高等院校计算机科学与技术“十五”规划教材

应 用 密 码 学

王衍波　薛　通　编著

机 械 工 业 出 版 社

VISIT…

LANZAROTE Caliente.com

出 版 说 明

信息技术高度普及的今天，具备一定层次的信息技术素养成为社会素质教育的一个重要目标，由此对高等院校的计算机专业教育提出了更高更新的要求。教育水平提高的关键是教学质量，那么对教学质量有直接影响的教材建设就成为了计算机专业教育的根本，为重中之重。

适逢高等院校计算机专业教育改革的关键时期，为配合相关的教材建设，机械工业出版社会同全国在该领域内享誉盛名、具备雄厚师资和技术力量的高等院校，包括清华大学、上海交通大学、南京大学、成都电子科技大学、东南大学、西安电子科技大学、解放军理工大学、北京科技大学等重点名校，组织了多位长期从事教学工作的骨干教师，集思广益，对当前高等院校的教学现状开展了广泛而深入的研讨，继而紧密结合当前技术发展需要并针对教学改革所提出的问题，精心编写了这套面向普通高等院校计算机专业的系列教材，并陆续出版。

本套教材内容覆盖了普通高等院校计算机专业学生的必修课程，另外还恰如其分地添加了一些选修课程，总体上分为基础、软件、硬件、网络和多媒体五大类。在编写过程中，对教学改革力度比较大、内容新颖以及各院校急需的并且适应社会经济发展的新教材，优先选择出版。

本套教材注重系统性、普及性和实用性，力求达到专业基础课教材概念清晰、深度合理的标准，并且注意与专业课教学的衔接；专业课教材覆盖面广、深浅适中，在体现相关领域最新发展的同时注重理论联系实际。全套教材体现了教育改革的最新思想，可作为高等院校计算机科学与技术专业的教学用书，同时也是培训班和自学使用的最佳教材。

机械工业出版社

前　言

随着计算机网络的普及，大量的电子数据通过网络传输到世界各地成为可能。但在具有重大的经济价值或关系国家、军队的命运等重要数据的传输过程中，任何一点泄露和差错都可能造成不可估量的损失。如何保证信息的机密性、真实性、不可否认性是密码学研究的重要课题。密码技术是信息安全的保障及核心技术。计算机网络、通信技术的发展和信息时代的到来，给密码学提供了难得的发展机遇，密码理论、密码技术、密码管理等研究与应用进入了一个新的时期。

现代密码技术被应用到了信息技术的所有领域，已从传统的单纯保密功能发展成为一门具有加密与密码分析、数字签名、信息鉴别、身份认证、密钥管理、安全协议等多分支的综合学科。

当前密码技术的研究领域非常广泛、深刻，主要集中在：

1．密码学的信息理论和计算机复杂性理论的研究，如密码中信息泄露的发现和利用、安全密码体制的准则和评测。

2．公开密钥密码理论研究，寻找可构作新的公开密钥算法的数学难题，RSA，椭圆曲线公钥密码算法等密码算法的快速实现。

3．对称密码理论研究，如对称密码的设计准则和评测等。

4．新型安全密码算法的研究，如量子密码、混沌密码理论、DNA 密码、信息隐藏技术等。

5．密码安全协议的研究，如数字签名、身份鉴别、数据完整性、密钥管理、秘密共享等设计理论和方法，虚拟专用网络技术，计算机网络通信安全技术等。

解放军理工大学于 1998 年开设信息安全与对抗方面的课程。本书是在相关讲义和 4 年的教学实践基础上形成的。我们本着务实、求真的目标，力图对密码学的主要理论和技术作比较详细的阐述，使学生不但能够掌握相应的密码学原理，而且能够将其应用于网络安全的实践中。

本书共 11 章，第 1、2 章讲解与现代密码学有关的数学基础知识。第 3 章叙述密码学的基本概念。因课时的限制，我们不准备对传统密码展开讨论，所以仅对传统密码算法作了一点简要介绍，使学生对传统密码不至于一无所知。第 4 章讲解对称密码算法，对于欧洲密码计划的候选算法，教学中可选择其中的一个讲解。第 5～7 章讲解公钥密码算法，数学签名和散列函数，序列密码学。第 8～10 章讲解现代密码学的三个新的重要方向：椭圆曲线密码学、混沌密码学、量子密码学。对已经成熟的椭圆曲线密码学，本书用较多的篇幅进行讲解，这一部分可作为选讲内容和学生毕业设计的阅读材料。第 11 章讲解密码协议。

本书的特点是：几乎对每一个著名的密码体制都给出了一个实用的例子，这是一般的密码学书籍所缺少的，这也是作者的教学体会。学生在编程实践时往往无法对结果进行检验，这些实用的例子使学生可以对自己编制的程序进行检验。

本书可以在 45～60 个理论学时内完成，再配以 15～20 个学时的编程实践。也可结合有关网络编程课程进行实验。

本书的写作得到了解放军理工大学通信工程学院电子信息工程系张雄伟教授、曹耀辉政委、肖军模教授的鼓励、支持和帮助。

在本书的写作过程中，刘爱琴同志作了大量的数据和文字审校工作；于杰山、赵小龙、端木庆峰阅读了本书的初稿，验证了大部分实例和习题；何俊、成必权、解萍同志也校对了部分文字。

作者感谢解放军理工大学通信工程学院教保科许晔峰、岳超，电子信息工程系信息技术教研室的刘军、杨健、任煜、冒俊峰等同志的支持和帮助。

作　者

2003 年 2 月

目　　录

出版说明
前言
第 1 章　初等数论基础 ······ 1
1.1　素数与因式分解 ······ 1
1.2　同余式理论 ······ 3
1.3　Euler 定理 ······ 7
1.4　平方剩余 ······ 9
1.5　素性检验与模幂算法 ······ 12
1.6　指数与原根 ······ 13
1.7　习题 ······ 16
1.8　实验题 ······ 17
第 2 章　近世代数基础 ······ 18
2.1　群论初步 ······ 18
2.2　域论初步 ······ 20
2.3　有限域中的计算 ······ 26
2.4　习题 ······ 29
2.5　实验题 ······ 29
第 3 章　密码学基本概念 ······ 30
3.1　密码技术发展简介 ······ 30
3.2　密码系统的概念 ······ 32
3.3　密码分析 ······ 33
3.4　数字签名与认证 ······ 34
3.5　计算复杂性理论 ······ 36
3.6　传统密码举例 ······ 37
3.7　习题 ······ 39
第 4 章　对称密码算法 ······ 41
4.1　美国数据加密标准 DES ······ 41
4.1.1　历史 ······ 41
4.1.2　算法 ······ 41
4.1.3　DES 实例 ······ 47
4.1.4　DES 的安全性 ······ 53
4.1.5　DES 工作方式 ······ 56
4.2　国际数据加密算法 IDEA ······ 58
4.2.1　IDEA 加密算法 ······ 59
4.2.2　IDEA 子密钥生成 ······ 60
4.2.3　IDEA 解密 ······ 61
4.2.4　IDEA 讨论 ······ 63

4.2.5 IDEA 实例 …… 64
4.3 美国高级数据加密标准 AES …… 66
4.3.1 算法 …… 66
4.3.2 基本运算 …… 67
4.3.3 基本部件 …… 69
4.3.4 AES 解密原理 …… 72
4.3.5 密钥生成器 …… 73
4.3.6 AES 实例 …… 75
4.4 欧洲密码标准 …… 76
4.4.1 引言 …… 76
4.4.2 Khazad …… 77
4.4.3 RC6 …… 80
4.4.4 Safer++加密算法 …… 81
4.5 习题 …… 90
4.6 实验题 …… 91
第 5 章 公钥密码算法 …… 92
5.1 引言 …… 92
5.1.1 公钥密码思想 …… 92
5.1.2 Diffie-Hellman 密钥交换协议 …… 93
5.2 RSA 密码体制 …… 93
5.2.1 体制描述 …… 94
5.2.2 算法分析 …… 94
5.2.3 安全性分析 …… 95
5.2.4 RSA 举例 …… 99
5.3 Rabin 公钥密码体制 …… 102
5.3.1 体制描述 …… 102
5.3.2 解密算法及原理 …… 102
5.3.3 安全性与因子分解 …… 103
5.3.4 错误明文攻击 …… 103
5.3.5 示例 …… 104
5.4 ElGamal 公钥密码体制 …… 104
5.4.1 体制描述 …… 104
5.4.2 示例 …… 105
5.5 MH 背包公钥密码体制 …… 105
5.5.1 0-1 背包问题 …… 105
5.5.2 超递增序列 …… 106
5.5.3 背包密码体制的原理 …… 106
5.5.4 MH 背包密码体制的描述 …… 107
5.5.5 示例 …… 108

5.6 概率公钥密码体制 ······ 109
5.6.1 引言 ······ 109
5.6.2 体制的描述 ······ 109
5.6.3 示例 ······ 110
5.7 NTRU 公钥密码体制 ······ 112
5.7.1 NTRU 的工作空间 ······ 112
5.7.2 密码体制 ······ 113
5.7.3 解密原理与参数选择 ······ 113
5.7.4 安全性分析 ······ 114
5.7.5 示例 ······ 116
5.8 公钥密码标准 ······ 116
5.8.1 PKCS #1 ······ 117
5.8.2 PKCS#3 v2.1 ······ 121
5.9 习题 ······ 122
5.10 实验题 ······ 124
第 6 章 数字签名方案与散列函数 ······ 126
6.1 数字签名方案 ······ 126
6.1.1 引言 ······ 126
6.1.2 RSA 签名方案 ······ 127
6.1.3 ElGamal 签名方案 ······ 128
6.1.4 数字签名标准 DSS ······ 131
6.1.5 其他数字签名方案 ······ 132
6.2 散列函数 ······ 135
6.2.1 签名与散列函数 ······ 135
6.2.2 无碰撞散列函数 ······ 136
6.2.3 离散对数散列函数 ······ 137
6.2.4 散列函数的扩展 ······ 139
6.3 散列函数标准 SHS ······ 140
6.3.1 MD5 ······ 140
6.3.2 SHA-1 ······ 144
6.3.3 SHA-256，SHA-384，SHA-512 ······ 148
6.3.4 MD5，SHA-1，SHA-256，SHA-384，SHA-512 的比较 ······ 151
6.4 习题 ······ 152
6.5 实验题 ······ 153
第 7 章 序列密码 ······ 154
7.1 序列密码模型 ······ 154
7.2 序列的随机性概念 ······ 155
7.3 线性反馈移位寄存器 ······ 157
7.4 m 序列及其随机性 ······ 159

7.5 周期序列的线性复杂度 ……162
7.6 习题 ……163
第 8 章 椭圆曲线密码学 ……165
8.1 引言 ……165
8.2 椭圆曲线的概念 ……166
8.2.1 概念和定义 ……166
8.2.2 群运算 ……167
8.2.3 椭圆曲线的分类 ……169
8.2.4 椭圆曲线分类加法公式 ……171
8.3 椭圆曲线群的结构 ……172
8.3.1 群的阶 ……172
8.3.2 椭圆曲线的同构类 ……173
8.4 有限域 F_2m 上的算术运算 ……175
8.4.1 F_2m 的基及其元素的的运算 ……175
8.4.2 有限域 F_2n 上椭圆曲线的倍点公式 ……177
8.5 椭圆曲线上的密码体制 ……180
8.5.1 Menezes-Vanstone 密码体制 ……182
8.5.2 F2m 上非超奇椭圆曲线 Menezes-Vanstone 密码算法 ……182
8.5.3 超奇椭圆曲线的密码体制 ……184
8.5.4 EDSA——椭圆曲线数字签名标准 ANSI X9.62 ……184
8.5.5 其他体制的椭圆曲线版本 ……185
8.6 习题 ……187
第 9 章 混沌理论在密码学中的应用 ……189
9.1 混沌的基本概念 ……189
9.2 混沌序列的产生及其随机序列 ……190
9.3 逆混沌密码体制 ……193
9.4 示例 ……193
9.5 实验题 ……194
第 10 章 量子密码理论 ……195
10.1 引言 ……195
10.2 Heisenberg 测不准原理 ……195
10.3 BB84 协议 ……196
10.3.1 无窃听量子信道的密钥分配过程 ……197
10.3.2 有窃听量子信道的密钥分配过程 ……198
10.4 习题 ……199
第 11 章 密码协议 ……200
11.1 引言 ……200
11.2 公证协议 ……200
11.2.1 仲裁协议 ……200

11.2.2 裁决协议 ······ 201
11.2.3 自动执行协议 ······ 201
11.3 密钥协议 ······ 201
11.3.1 密钥交换协议 ······ 202
11.3.2 密钥协商协议 ······ 203
11.4 秘密共享 ······ 203
11.4.1 Shamir 门限方案 ······ 204
11.4.2 Blakley 秘密共享方案 ······ 206
11.4.3 带有不同权限的秘密共享方案 ······ 206
11.4.4 秘密共享中的欺骗 ······ 207
11.5 网络游戏 ······ 207
11.5.1 扑克游戏 ······ 207
11.5.2 电子掷币协议 ······ 209
11.6 电子投票协议 ······ 210
11.7 习题 ······ 211
11.8 实验题 ······ 211
参考文献 ······ 212

第1章　初等数论基础

1.1　素数与因式分解

记号：除特别注明外，英文字母 a，b，c，d，m，n 等都表示正整数。

定义 1-1　如果 a 可以除尽 b，则称 a 整除 b，记为：$a|b$。

如果 $a|b$，$a|c$，则称 a 是 b，c 的一个公因子；如果 a 是 b，c 的公因子中最大者，则称 a 是 b，c 的最大公因子，记为：a=gcd(b,c)，或 a=(b,c)。gcd 是 greatest common divisor 的缩写。

可以证明，b，c 的每一个公因子都可以整除 a。

【例 1-1】　(4,6)=2，(15,70)=5。

定义 1-2　如果 $a|c$，则称 c 是 a 的倍数；如 $a|c$，$b|c$，则称 c 是 a，b 的公倍数。

如果 c 是 a，b 的公倍数中最小者，则称 c 是 a，b 的最小公倍数，记为：c=lcm{a,b}，或 c=[a,b]。lcm 是 least common multiple 的缩写。

可以证明，c 可以整除 a，b 的任一公倍数。

【例 1-2】　lcm{15,20,30}=60。

定义 1-3　除 1 外，只能被 1 和其本身整除的自然数称为素数，不是 1，且非素数的正整数称为合数。

例如 2，3，5，7，11，13 等都是素数；4，6，8，9，10 等都是合数。

引理　若 $a=qb+r$，$0\leqslant r<b$，则$(a,b)=(b,r)$。

证明　令 $d=(a,b)$，$d'=(b,r)$，则 $d\mid r=a-qb$，$\Rightarrow d\mid(b,r)=d'$，得 $d\leqslant d'$；另一方面，$d'\mid a=qb+r$，$\Rightarrow d'\mid(a,b)=d$，得：$d'\leqslant d$，所以 $d=d'$。

定理 1-1　若 $d=(a,b)$，则存在整数 p，q，使得 $d=pa+qb$。

证明　不妨设 $a>b$。应用一系列简单的除法（称为辗转相除法）可得：

$a=q_0b+r_0$　　$0\leqslant r_0<b$

$b=q_1r_0+r_1$　　$0\leqslant r_1<r_0$

$r_0=q_2r_1+r_2$　　$0\leqslant r_2<r_1$

……………　　…………

$r_{k-2}=q_kr_{k-1}+r_k$　　$0\leqslant r_k<r_{k-1}$

$r_{k-1}=q_{k+1}r_k$

依次将 $r_k, r_{k-1},\ldots, r_1, r_0$ 后一式回代入前一式，可得：$r_k=pa+qb$。

再由引理得：$(a,b)=(b,r_0)=(r_0,r_1)=\cdots=(r_{k-2},r_{k-1})=(r_{k-1},r_k)=r_k$。

所以，$d=(a,b)=r_k$。定理证毕。

这个定理的证明也给出了最大公因子的求法——辗转相除法。

【例 1-3】 求(45,39)。

解：45=39×1+6

39=6×6+3

6=3×2

所以，(45,39)=3。

【例 1-4】 求p，q使（124，29）=124p+29q。

解：124=3×29+8

29=3×8+5

8=1×5+3

3=1×2+1

2=1×1

所以(124,39)=1。

回代：

1=3-1×2

=3-1×(5-1×3)=2×3-1×5

=2×(8-1×5)-1×5=2×8-3×5

=2×8-3×(29-3×8)=11×8-3×29

=11×(124-3×29)-3×29=11×124-36×29

所以有：1=11×124+（-36）×29。

定理 1-2 （数的表示定理）每一个正合数，都可表示为素数的乘积。不考虑乘积顺序时，表示法是惟一的。即：对任意$n\in N$，有：

$$n=p_1^{s_1}p_2^{s_2}\cdots p_t^{s_t},$$

其中p_1，p_2,…，p_t是互不相同的素数，s_1，s_2,…，s_t是正整数。

证明 若n是合数，则存在a，b，使得$n=ab$；若a，b是合数，则又有$a=ed$，$b=gf$；

继续以上步骤，直到不能分解为止。

这个过程在有限步内完成（否则，n将变成无穷大），故有：$n=g_1g_2\cdots g_k$，其中$g_i,i=1,,2,\cdots,l$都是素数，将相同素数写成指数形式即得合数的标准分解式。

再证惟一性：

设有两个分解式：$n=g_1g_2\cdots g_k$，$n=h_1h_2\cdots k_l$，则：$h_1h_2\cdots h_l=g_1g_2\cdots g_k$，推得：$h_1\mid g_1g_2\cdots g_k$。

由于h_1，g_1，g_2,…，g_k都是素数，所以，存在g_i，$h_1=g_i$。

不失一般性，不妨设$h_1=g_1$。

在等式两边消去$h_1=g_1$，则等式成为：$h_2h_3\cdots h_l=g_2g_3\cdots g_k$。继续进行同样的步骤，直到$h_l$，可知必有$l=k$，$h_l=g_l$。定理证毕。

定理 1-3 设$a \geq 2$是正整数，则：

（1）如果a是合数，那么必存在素数$p \leq \sqrt{a}$，$p|a$；

（2）如果有素数分解式$a = p_1 \cdots p_s$，则存在素数$p \leq \sqrt[s]{a}$，$p|a$。

证明 （1）如果整除a的素数都大于$\sqrt{a}$，则因a是合数，所以至少有两个素数$p,q > \sqrt{a}$，$p|a$，$q|a$，从而有$a \geq pq > \sqrt{a}\sqrt{a} = a$，矛盾；

（2）如果结论不成立，则$p_i > \sqrt[s]{a}, i = 1,2,\cdots,s$，得到$a = p_1 \cdots p_s > \underbrace{\sqrt[s]{a} \cdots \sqrt[s]{a}}_{s个} = a$，矛盾，所以结论成立。定理证毕。

根据定理 1-3 可以得到一种寻找素数的有效方法：

Eratoschenes 筛法：要求不超过N的所有素数，只要将 1 和不超过N的合数去掉即可。

而由定理 1-3（1），这些合数必有素因子$p \leq \sqrt{N}$，因而，如果先求出$\leq \sqrt{N}$的素数：$p_1,\cdots,p_k$，然后用这些素数试除$2,\cdots,N$，则不被$p_1,\cdots,p_k$整除的就是所有素数。

进一步，当得到了$2,\cdots,N$中的所有素数$p_1,\cdots,p_k$后，又可以用其检测 2，…，N，…，N^2之中的素数了，这一过程继续下去，即可筛出所有素数。

例如要求不超过N的所有素数。从素数 2 开始，筛去$2,\cdots,N$中 2 的倍数；那么，剩下数中$< 2^2 = 4$的数 3 就是素数；再用 2，3 去筛后面剩下的数，但 2 已经用过，所以只须用 3 去筛，即筛去 3 的倍数；剩下的数中$< 3^2 = 9$的都是素数。即 2，3，5，7。所以，进一步可以用 5，7 去筛，得到素数 11，13，17，19，23，29，37，41，43，47；依此类推，即可得到所求素数。求 100 之内的所有素数如图 1-1 所示。

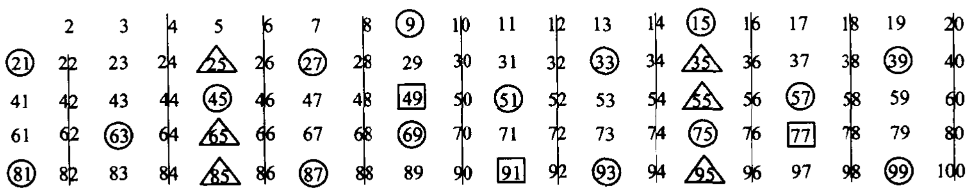


图 1-1　100 之内的筛法

可见，所求素数为：2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,83,89,97。

求大素数是数学中的一个难题。密码学应用中，不但需要检测一个整数是否为素数，而且要求计算速度快。进一步，对素数还有更高的要求，即所谓的强素数、安全素数。素性检验和强素数的求取等等都是密码学的重要研究课题，我们将在后面的具体算法中涉及时讲解。

定理 1-4 素数有无穷多个。

证明 用反证法。设有有限个素数$p_1,\cdots,p_n$，令$a = p_1 \cdots p_n + 1$，易见$a > 2$，$p_i \nmid a$，$i = 1,2,\cdots,n$。但是，根据定理 1-3，应有$p_i | a$，矛盾。所以，有无穷多个素数。定理证毕。

1.2 同余式理论

定义 1-4 设m是正整数，如果$m|a-b$，即：$a - b = km$，则称a和b模m同余，记

为：$a \equiv b \bmod m$。m 称为这个同余式的摸。

定理 1-5 模 m 的同余关系是等价关系，即满足：

（1）自反性：$a \equiv a \bmod m$；

（2）对称性：若 $a \equiv b \bmod m$，则 $b \equiv a \bmod m$；

（3）传递性：若 $a \equiv b \bmod m$，$b \equiv c \bmod m$，则 $a \equiv c \bmod m$。

证明简单，略去。

定理 1-6 若 $a \equiv b \bmod m$，$c \equiv d \bmod m$，那么：

（1）$a \pm c \equiv b \pm d \bmod m$；

（2）$ac \equiv bd \bmod m$；

（3）若 $c \mid a$, $c \mid b$, $(c,m)=k$，则 $\frac{a}{c} \equiv \frac{b}{c} \bmod \frac{m}{k}$；特别地，如果 $(c,m)=1$，则 $\frac{a}{c} \equiv \frac{b}{c} \bmod m$，即同余式两边可以同除以一个与 m 互素的数 c。

（1），（2）的证明是简单的，我们只证（3）。

证明（3）：设 $a=a'c$，$b=b'c$，$c=c'k$，$m=m'k$，$(c',m')=1$。

则由：$a \equiv b \bmod m$

$\Rightarrow$ $a'c \equiv b'c \bmod m'k$，即：$a'c-b'c=lkm'$，

$\Rightarrow$ $c \mid lkm'$，即：$kc' \mid lkm'$，

$\Rightarrow$ $c' \mid lm'$，但是，$(c',m')=1$，

$\Rightarrow$ $c' \mid l$。

$\Rightarrow$ $a'-b'=\frac{l}{c'} m'$，即 $\frac{a}{c} \equiv \frac{b}{c} \bmod \frac{m}{(c,m)}$。定理证毕。

定义 1-5 $ax \equiv b \bmod m$ 称为线性同余式，满足同余式的整数 x 称为同余式的解。

可以证明，如果 x_1 是该线性同余式的解，则对于任意 $x_2 \equiv x_1 \bmod m$ 也是该同余式的解，但是，反过来却不正确，可见下面的定理 1-7（2）。

定理 1-7 设 $d=(a,m)$，则对于同余式 $ax \equiv b \bmod m$，

（1）有解的充要条件是：$d \mid b$；

（2）如果 x_0 是同余式的一个解，那么同余式的所有解为：$x \equiv x_0 \bmod \frac{m}{d}$。

证明 （1）设 x_0 是同余式的一个解，则存在整数 k，$ax_0-km=b$，于是得：

$$d(\frac{a}{d}x_0-\frac{m}{d}k)=b，即得：d \mid b。$$

反之，若 $d \mid b$，令 $b=b'd$，$a=a'd$，$m=m'd$，则 $(a',m)=1$，由定理 1-1 知，存在整数 p，q，使得：$pa'+qm'=1$，于是：

$$\begin{aligned} b &= bpa'+bqm' \\ &= b'dpa'+b'dqm' \\ &= pb'(a'd)+qb'(m'd) \\ &= pb'a+qb'm \end{aligned}$$

$\Rightarrow$ $a(pb')=b-(qb')m$，即：$a(pb') \equiv b \bmod m$，

所以，pb'是同余式的解。

（2）如果x_0是同余式的解，则易验证x_0+km'也是同余式的解。

反之，若x_0'是同余式的任意一个解，则有：

$ax_0 \equiv b \bmod m$，$ax_0' \equiv b \bmod m$，得到：$ax_0 \equiv ax_0' \bmod m$，

根据定理 1-6（3）得到：$x_0 \equiv x_0' \mod \dfrac{m}{d}$。定理证毕。

定义 1-6 若$ab \equiv 1 \mod m$，则称b是a模m的逆元。

显然a也是b模m的逆元。

定理 1-8 如果$(a,m)=1$，则a模m的逆元一定存在。

证明 由于$(a,m)=1$，所以，由定理 1-1 知，

存在整数s,t，使得$sa+tm=(a,m)=1$，于是，$as \equiv 1 \mod m$。

记$\tilde{s}=s+km>0$，那么$a\tilde{s}=as+akm \equiv 1 \mod m$，正整数$\tilde{s}$就是$a$模$m$的逆元。定理证毕。

但是，s,t是用回代的方法得到的，因此求解时必须记住每一个中间值，用作回代时使用，这需要巨大的空间开销，显然这是不可取的，不适合密码应用。

稍作观察可以发现，采用从前向后代入的方法，可以避免存储空间问题。例如，例 1-5 可进行如下：

124=3×29+8

　　8=124−3×29

29=3×8+5

　　5=29−3×8=29-3×(124-3×29)= −3×124+10×29

8=1×5+3

　　3=8−1×5=124−3×29−1×(−3×124+10×29)=4×124−13×29

5=1×3+2

　　2=5−1×3= −3×124+10×29−1×(4×124−13×29)= −7×124+23×29

3=1×2+1

　　1=3−1×2=4×124−13×29−1×(−7×124+23×29)=11×124+36×29。

可见，这个方法是可行、有效的。下面的定理，给出这一方法的紧凑形式。

定理 1-9 设$r_1 \equiv b_1u \mod m$，$r_2 \equiv b_2u \mod m$，$r_1=qr_2+r_3$，则 $r_3 \equiv (b_1-qb_2)u \mod m$。

证明 $r_1=b_1u+lm$，$r_2=b_2u+km$，

$r_3=r_1-qr_2=b_1u+lm-q(b_2u+km)=(b_1-qb_2)u+(l-qk)m$，所以

$r_3 \equiv (b_1-qb_2)u \mod m$。定理证毕。

求模逆算法：设$0<u<m$，$(u,m)=1$，求u模m的逆元，即：求x，$0<x<m$，$ux\equiv1 \bmod m$。

（1）令$r_1=m$，$r_2=u$，则：$r_1=m\equiv 0\cdot u \mod m$，$r_2=u\equiv 1\cdot u \mod m$，即有：$b_1=0$ $b_2=1$。

（2）令$r_1=q_1r_2+r_3$，$0\leqslant r_3<r_2$，得到$r_3\equiv(b_1-q_1b_2)u \mod m$，记$b_3=b_1-q_1b_2$，则

得到$r_2 \equiv b_2 u \mod m$，$r_3 \equiv b_3 u \mod m$。

（3）令$r_2 = q_2 r_3 + r_4, 0 \leqslant r_4 < r_3$，得到$r_4 \equiv (b_2 - q_2 b_3) u \mod m$，记$b_4 = b_2 - q_2 b_3$，则得到$r_4 \equiv b_4 u \mod m$。

（4）继续下去，总之有：

$r_1 \equiv b_1 u \mod m$，$r_2 \equiv b_2 u \mod m$，$r_1 = q_1 r_2 + r_3$，$0 \leqslant r_3 < r_2$，

$r_2 \equiv b_2 u \mod m$，$r_3 \equiv b_3 u \mod m$，$r_2 = q_2 r_3 + r_4$，$0 \leqslant r_4 < r_3$，

……………………

$r_{k-1} \equiv b_{k-1} u \mod m$，$r_k \equiv b_k u \mod m$，$r_{k-1} = q_{k-1} r_k + r_{k+1}$，$0 \leqslant r_{k+1} < r_k$，

……………………

由于$r_2 > r_3 > \cdots > r_k > r_{k+1} \cdots \geqslant 0$，所以，以上操作必终止于有限步，不妨设$r_{k+1} = 0$，那么必有$1 = r_k = (r_k, r_{k-1}) = \cdots = (r_3, r_2) = (r_2, r_1) = (u, m)$。

所以由$r_k \equiv b_k u \mod m$，得$b_k u \equiv 1 \mod m$，于是所求为：$u^{-1} \equiv b_k \mod m$。

根据以上过程，不难给出模逆算法的流程图 1-2：

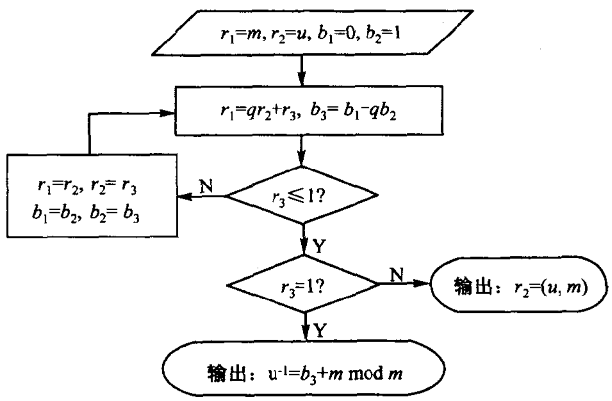


图 1-2 模逆求解流程图

定理 1-10 （中国剩余定理 CRT）设$m_1, m_2, \cdots, m_k$是两两互素的正整数，即：$(m_i, m_j) = 1$，$i \neq j$，$i, j = 1,2,\cdots,k$，则同余方程组：

$$\begin{cases} x \equiv b_1 \mod m_1 \\ x \equiv b_2 \mod m_2 \\ \cdots\cdots\cdots\cdots \\ x \equiv b_k \mod m_k \end{cases}$$

模$[m_1, m_2, \cdots, m_k]$有惟一解，即在模$[m_1, m_2, \cdots, m_k]$的意义下，存在惟一的x，满足：

$$x \equiv b_i \mod [m_1, m_2, \cdots, m_k]，i = 1,2,\cdots,m。$$

证明 令$M = m_1 m_2 \cdots m_k$，$M_i = \dfrac{M}{m_i}$。

由于$(M_i, m_i) = 1$，所以，由定理 1-8 存在y_i，使得：$y_i M_i \equiv 1 \mod m_i$。

令 $x = b_1M_1y_1 + b_2M_2y_2 + \cdots + b_kM_ky_k$，下面证明，此 x 就是所求解。

由于 $j \neq h$ 时，$m_j \mid M_h$，所以，$M_h \equiv 0 \bmod m_j$。

于是，x 中除第 j 项外，其余各项模 m_j 都为零，从而有：

$$x \equiv b_jM_jy_j \bmod m_j$$
$$\equiv b_j \bmod m_j$$

再证 $x = b_1M_1y_1 + b_2M_2y_2 + \cdots + b_kM_ky_k$ 是惟一解。

设 $\bar{x}$ 是另一个解，则 $x \equiv \bar{x} \equiv b_j \bmod m_j\ j = 1,2,\cdots,k$。

$\Rightarrow$ $x - \bar{x} \equiv 0 \bmod m_j$，于是，$m_j \mid (x - \bar{x})$，$j=1,2,\ldots,k$。

$\Rightarrow$ $M \mid (x - \bar{x})$，或 $x \equiv \bar{x} \bmod M$。

所以在模 $M = [m_1, m_2, \cdots, m_k]$ 的意义下同余式组的解是惟一的。

【例 1-5】 求解同余式组：$x \equiv 2 \bmod 5$，$x \equiv 4 \bmod 7$，$x \equiv 3 \bmod 9$。

解 $M = 5 \times 7 \times 9 = 315$；$M_1 = 63$，$M_2 = 45$，$M_3 = 35$；

$$63y_1 \equiv 1 \bmod 5, \Rightarrow y_1 = 2;$$
$$45y_2 \equiv 1 \bmod 7, \Rightarrow y_2 = 5;$$
$$35y_3 \equiv 1 \bmod 9, \Rightarrow y_3 = 8。$$

所求解为：$x = 2 \times 63 \times 2 + 4 \times 45 \times 5 + 3 \times 35 \times 8 = 1992 \equiv 102 \bmod 315$。

1.3 Euler 定理

所有模 m 和 r 同余的整数组成一个类 $[r]$ 称为与 r 同余的剩余类。0，1，2，…，m-1 是其一个完备代表元系，也叫完全剩余系。

定义 1-7 记与 m 互素的剩余类的个数为 $\varphi(m)$，称为 m 的 Euler 函数。

定理 1-11 若 $(m_1, m_2) = 1$，则 $\varphi(m_1m_2) = \varphi(m_1)\varphi(m_2)$。

证明 （1）设 x 是一个正整数，$0 \leqslant x < m_1m_2$。

令 $x \equiv r_1 \bmod m_1$，$x \equiv r_2 \bmod m_2$，则 x 惟一地确定了 (r_1, r_2)，其中 $0 \leqslant r_1 < m_1$，$0 \leqslant r_2 < m_2$。

反之，对于任意给定的数对 (r_1, r_2)，$0 \leqslant r_1 < m_1$，$0 \leqslant r_2 < m_2$，令：

$$\begin{cases} x \equiv r_1 \bmod m_1 \\ x \equiv r_2 \bmod m_2 \end{cases}$$

由中国剩余定理可以得到惟一解：$x \bmod m_1m_2$。

所以，x 与数对 (r_1, r_2) 一一对应。

（2）x 与 m_1m_2 互素 $\Leftrightarrow$ x 与 m_1，m_2 都互素 $\Leftrightarrow$ r_1 与 m_1 互素，且 r_2 与 m_2 互素。

可见，与 m_1m_2 互素的数 x 的个数等于数对 $\{(r_1, r_2), (r_1, m_1) = 1, (r_2, m_2) = 1\}$ 的个数。

比 m_1 小，与 m_1 互素的 r_1 的个数为 $\varphi(m_1)$，比 m_2 小，与 m_2 互素的 r_2 的个数为 $\varphi(m_2)$，所以：数对 $\{(r_1, r_2), (r_1, m_1) = 1, (r_2, m_2) = 1\}$ 的个数为 $\phi(m_1)\phi(m_2)$，即得：

$\varphi(m_1m_2) = \varphi(m_1)\varphi(m_2)$。定理证毕。

定理 1-12 设 $m = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$ 是 m 的标准分解式，$p_1, p_2, \cdots, p_k$ 是互不相同的素数，

则：$\varphi(m)=m(1-\frac{1}{p_1})(1-\frac{1}{p_2})\cdots(1-\frac{1}{p_k})$。

证明 （1）首先证明$\varphi(p^\alpha)=p^\alpha(1-\frac{1}{p})$。

在区间$[0,p^\alpha)$上的正整数，或是p的倍数，或与p^α互素。而p的倍数为：$0,p,2p,\cdots,(p^{\alpha-1}-1)p$，共$p^{\alpha-1}-1$个。可见，和$p^\alpha$互素的数的个数是：

$(p^\alpha-1)-(p^{\alpha-1}-1)=p^\alpha-p^{\alpha-1}$，

所以：$\varphi(p^\alpha)=p^\alpha(1-\frac{1}{p})$。

（2）由定理 1-11 得，$\varphi(m)=\varphi\left(p_1^{\alpha1}\ p_2^{\alpha2}\cdots p_k^{\alpha k}\right)=\varphi\left(p_1^{\alpha1}\right)\cdots\varphi\left(p_2^{\alpha2}\ \right)$，易得：

$\varphi(m)=m(1-\frac{1}{p_1})(1-\frac{1}{p_2})\cdots(1-\frac{1}{p_k})$。定理证毕。

定理 1-13 （Euler 定理）若$(a,m)=1$，则$a^{\varphi(m)}\equiv 1 \bmod m$。

证明 设$\varphi(m)=k$，并令$r_1,r_2,\cdots,r_k$是与m互素的模m的全体剩余类代表元。

由于$(a,m)=1$，所以$ar_1,ar_2,\cdots,ar_k$也是与m互素的，且两两不同余，也是与m互素的模m的全体剩余类代表元。所以，

$$a^k r_1 r_2\cdots r_k=ar_1 ar_2\cdots ar_k\equiv r_1 r_2\cdots r_k \bmod m,$$

由于$r_1,r_2,\cdots,r_k$与m互素，所以$(r_1r_2\cdots r_k,m)=1$，以上同余式两边消去公因子得：$a^k\equiv 1 \bmod m$。定理证毕。

由定理可知，若$(a,m)=1$，则相对于模m，a有逆元是$a^{\varphi(m)-1}$，即：

$a^{\varphi(m)-1}\cdot a\equiv 1 \bmod m$。

于是有：

推论：对于同余式$ax\equiv b \bmod m$，有：

（1）若$(a,m)=1$，则：$x\equiv a^{\varphi(m)-1}b \bmod m$；

（2）若$(a,m)=d$，且$d\mid b$（如果$d\nmid b$，则同余式无解），则：

$$x\equiv\left(\frac{a}{d}\right)^{\phi\left(\frac{m}{d}\right)-1}\cdot\left(\frac{b}{d}\right) \bmod \frac{m}{d}。$$

证明：（1）若$(a,m)=1$，则：

$x\equiv a^{\phi(m)}x=a^{\phi(m)-1}ax\equiv a^{\phi(m)-1}b \bmod m$，即：

$x\equiv a^{\varphi(m)-1}b \bmod m$。

（2）若$(a,m)=d$，且$d\mid b$，则：

$ax\equiv b \bmod m$与$\frac{a}{d}x\equiv\frac{b}{d}\bmod\frac{m}{d}$同解，这时$(\frac{a}{d},\frac{m}{d})=1$，所以，$x\equiv\left(\frac{a}{d}\right)^{\phi\left(\frac{m}{d}\right)-1}\cdot\left(\frac{b}{d}\right)\bmod\frac{m}{d}$。

【例 1-6】 求解同余方程：$11x\equiv 23 \bmod 42$。

解 $\because(11,42)=1$,

$\therefore x \equiv 11^{\phi(42)-1} \cdot 23 \bmod 42$

$\equiv 11^{\phi(6)\phi(7)-1} \cdot 23 \bmod 42$

$\equiv 11^{5\times 6-1} \cdot 23 \bmod 42$

$\equiv 11^{29} \cdot 23 \bmod 42$

$\equiv 25 \bmod 42$。

作为 Euler 定理的推论有：

定理 1-14 （Fermat 定理）若 p 是素数，则 $a^p \equiv a \bmod p$。

定理 1-15 （Wilson 定理） p 是素数的充要条件是：$(p-1)! \equiv -1 \bmod p$。

证明 必要性：

$p=2$时，易证定理为真。

$p>2$ 时，由定理 1.2.4 知，对任意整数 a:$1 \leqslant a < p$，存在逆元 $\bar{a}$，$0<\bar{a}<p$，$a\bar{a} \equiv 1 \bmod p$。

因为：$a^2 \equiv aa \equiv 1 \bmod p \Leftrightarrow p \mid (a^2-1) \Leftrightarrow p \mid (a-1)$ 或 $p \mid (a+1)$

$\Leftrightarrow a \equiv 1 \bmod p$ 或 $a \equiv -1 \bmod p$。

所以，$1,2,3,\cdots,p-2,p-1$中，除$1,p-1 \equiv -1 \bmod p$外都不以自身为逆，从而$2,3,\cdots$，$p-2$这$p-3$个数，可分两两配对，分成个数相同的两个组，所以有：

$2 \cdot 3 \cdot \cdots \cdot (p-2) \equiv 1 \bmod p$。

于是：$(p-1)! = 1 \cdot 2 \cdot 3 \cdot \cdots \cdot (p-2) \cdot (p-1) \equiv 1 \cdot (p-1) \equiv -1 \bmod p$。

即得：$(p-1)! \equiv -1 \bmod p$。

充分性：

设$(p-1)! \equiv -1 \bmod p$，$p=ab$，$1<a<p, 1<b<p$。

因$a<p$，所以：$a \mid (p-1)!$，

再由假定$(p-1)! \equiv -1 \bmod p$，

$\Rightarrow (p-1)!+1 \equiv 0 \bmod p$，所以：$p \mid [(p-1)!+1]$，

$\Rightarrow a \mid [(p-1)!+1]$。

$\Rightarrow a \mid [(p-1)!+1]-(p-1)! = 1$，

$\Rightarrow a \mid 1$，

此与$a>1$矛盾，所以p是素数。定理证毕。

1.4 平方剩余

定义 1-8 设p是奇素数，$(a,p)=1$，若存在整数x满足：$x^2 \equiv a \bmod p$，则称a是模p的平方剩余，否则称a为非平方剩余。

易见，若x_0是$x^2 \equiv a \bmod p$的解，则$x \equiv x_0 \bmod p$也是a的二次剩余。

引理 如果p是奇素数，$p \nmid a$，则：

$$x^2 \equiv a \bmod p,$$

无解或有两个模p不同的解。

证明 （1）如果 x 是 $x^2 \equiv a \bmod p$ 的解，$x<p$，易证 $-x$ 也是其解。

（2）$-x \equiv x \bmod p \Leftrightarrow 2x \equiv 0 \bmod p$

$\Leftrightarrow p \mid 2$ 或 $p \mid x$（$\Leftrightarrow p \mid a$），这是不可能的，

所以：$-x \not\equiv x \bmod p$。

（3）设 $x'^2 \equiv a \equiv x''^2 \bmod p$，

$\Rightarrow x'^2 - x''^2 = (x'-x'')(x'+x'') \equiv 0 \bmod p$，

$\Rightarrow p \mid (x'-x'')$ 或 $p \mid (x'+x'')$，

$\Rightarrow x' \equiv x'' \bmod p$ 或 $x' \equiv -x'' \bmod p$。定理证毕。

定理 1-16 若 p 是奇素数，则整数 $1,2,\cdots,p-1$ 中恰有 $\frac{p-1}{2}$ 是模 p 平方剩余，$\frac{p-1}{2}$ 个是非平方剩余。

证明 （1）显然 $1^2,2^2,\cdots,(\frac{p-1}{2})^2$ 是模 p 不同余的平方剩余，共有 $\frac{p-1}{2}$ 个。

（2）对任意一个模 p 的平方剩余 a，它的两个平方根 x_1，$p-x_1$ 必有一个在 $1,2,\cdots,\frac{p-1}{2}$ 中。所以，模 p 的平方剩余是而且只是：$1^2,2^2,\cdots,(\frac{p-1}{2})^2$。

所以，非平方剩余有 $p-1-\frac{p-1}{2}=\frac{p-1}{2}$ 个。定理证毕。

定义 1-9 令 p 是素数，a 是一整数，$p \nmid a$，引进符号 $\left(\frac{a}{p}\right)$：

$$\left(\frac{a}{p}\right)=\begin{cases}1, & a\text{ 是模 }p\text{ 平方剩余；}\\ -1, & a\text{ 模 }p\text{ 非平方剩余。}\end{cases}$$

称为 Legendre 符号。

定理 1-17 （Euler 定理）设 p 是一个奇素数，a 是一整数，$p \nmid a$，则：

$\left(\frac{a}{p}\right) \equiv a^{\frac{p-1}{2}} \bmod p$。

证明 （1）若 $\left(\frac{a}{p}\right)=1$，即 $x^2 \equiv a \bmod p$ 有解，设其为 x，由 Fermat 定理得：

$$a^{\frac{p-1}{2}} \equiv (x^2)^{\frac{p-1}{2}} \equiv (x)^{p-1} \equiv 1 \bmod p\text{，从而 }\left(\frac{a}{p}\right) \equiv a^{\frac{p-1}{2}} \bmod p\text{。}$$

（2）若 $\left(\frac{a}{p}\right)=-1$，则 $x^2 \equiv a \bmod p$ 无解。由于 $(a,p)=1$，所以对任意 $1 \leqslant i \leqslant p-1$，总存在 j，使得：

$ij \equiv a \bmod p$，（因为 $(i,p)=1$，所以 $ix \equiv a \bmod p$ 有解），由于 $x^2 \equiv a \bmod p$ 无解，所以 $i \neq j$。于是，$1,2,3,\cdots,p-1$ 可分成 $\frac{p-1}{2}$ 对，每对之积模 p 都为 a。

因此：$(p-1)! \equiv a^{\frac{p-1}{2}} \bmod p$。所以，由 Wilson 定理得：

$\left(\frac{a}{p}\right) \equiv a^{\frac{p-1}{2}} \bmod p$。定理证毕。

定理 1-18 若 p 是奇素数，$p \nmid ab$，则：

（1）若 $a \equiv b \bmod p$，则 $\left(\frac{a}{p}\right)=\left(\frac{b}{p}\right)$；

（2）$\left(\frac{ab}{p}\right)=\left(\frac{a}{p}\right)\left(\frac{b}{p}\right)$；

（3）$\left(\frac{a^2}{p}\right)=1$。

证明 （1）$a \equiv b \bmod p$，

$\Rightarrow x^2 \equiv a \bmod p$ 有解 $\Leftrightarrow x^2 \equiv a \equiv b \bmod p$ 有解，所以：$\left(\frac{a}{p}\right)=\left(\frac{b}{p}\right)$。

（2）由 Euler 定理知：

$\left(\frac{a}{p}\right) \equiv a^{\frac{p-1}{2}} \bmod p$，$\left(\frac{b}{p}\right) \equiv b^{\frac{p-1}{2}} \bmod p$，$\left(\frac{ab}{p}\right) \equiv (ab)^{\frac{p-1}{2}} \bmod p$，

$\Rightarrow \left(\frac{a}{p}\right)\left(\frac{b}{p}\right) \equiv a^{\frac{p-1}{2}} b^{\frac{p-1}{2}} = (ab)^{\frac{p-1}{2}} \equiv \left(\frac{ab}{p}\right) \bmod p$，

但 Legendre 符号的值只能是 ±1，所以：$\left(\frac{ab}{p}\right)=\left(\frac{a}{p}\right)\left(\frac{b}{p}\right)$。定理证毕。

（3）因 $\left(\frac{a}{p}\right)=\pm 1$，所以由（2）可得：$\left(\frac{a^2}{p}\right)=\left(\frac{a}{p}\right)\left(\frac{a}{p}\right)=1$。

定理 1-19 （互逆定律）设 p，q 是两个奇素数，且 $p \neq q$，则：$\left(\frac{q}{p}\right)=(-1)^{\frac{p-1}{2}\cdot\frac{q-1}{2}}\left(\frac{p}{q}\right)$。

证明略去。

定义 1-10 设 n 是正整数，且有标准分解式 $n=p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$，则定义 Jacobi 符号为：

$$\left(\frac{a}{n}\right)=\left(\frac{a}{p_1}\right)^{\alpha_1}\left(\frac{a}{p_2}\right)^{\alpha_2}\cdots\left(\frac{a}{p_k}\right)^{\alpha_k},$$

左边为 Legendre 符号，这是 Legendre 符号的一种推广。

当 n 是素数时，Jacobi 符号就是 Legendre 符号，但当 n 是合数时，Jacobi 符号不能成为 $x^2 \equiv a \bmod n$ 是否有解的判据。

定理 1-20 设 n 是正合数，a，b 是与 n 互素的整数，即：$(ab,n)=1$，则：

（1）若 $a \equiv b \bmod n$，则 $\left(\frac{a}{n}\right)=\left(\frac{b}{n}\right)$；

（2）$\left(\frac{ab}{n}\right)=\left(\frac{a}{n}\right)\left(\frac{b}{n}\right)$；

（3）$\left(\frac{1}{n}\right)=1, \left(\frac{-1}{n}\right)=1, \left(\frac{2}{n}\right)=(-1)^{\frac{n^2-1}{8}}$；

（4）$\left(\frac{m}{n}\right)=(-1)^{\frac{n-1}{2}\times\frac{m-1}{2}}\left(\frac{n}{m}\right)$。

证明很简单，略去。

1.5 素性检验与模幂算法

定义 1-11 判别一个整数是不是素数的过程称为素性检验。

常用方法有：

（1）Wilson 定理：若 $(n-1)!\equiv -1 \bmod n$，则 n 是素数；

（2）Eratoschenes 筛法：若 $\sqrt{n}$ 不是整数，且 n 不能被任何小于 $\sqrt{n}$ 的正整数整除，则 n 是素数；

（3）概率算法（Solovay&Strassen）：若要检测整数 n 的素性，在 $1,2,\cdots,n-1$ 中随机地选出 100 个数 $a_i\ (1\leqslant i\leqslant 100)$，如果对每个 a_i，$(a_i,n)=1$，（素数的必要条件）$\left(\frac{a_i}{n}\right)\equiv a_i^{\frac{n-1}{2}} \bmod n$，（素数的必要条件）成立，则 n 是素数的概率是 $1-\frac{1}{2^{100}}$，所以我们可以认为 n 是素数。

从前面的内容可见，幂模运算是一种经常性的操作。在以后讲解的公钥密码算法中可以看到，幂模运算的速度是公钥密码算法的一项重要指标，从而研究快速有效的幂模算法是一件必须的工作，常用的幂模算法有：

（1）先作幂再作模：$a^m \bmod n$。显然，这种算法对稍大一点的数，所需空间开销就会超出机器的承受能力，一般不可能实现。这种算法需要 m 次乘法，一次除法。

（2）逐次乘模：$\underbrace{(\cdots(((a \bmod n)a \bmod n)a \bmod n)a\cdots)a}_{m次} \bmod n$。这种算法避免了空间开销问题，但是需要 m 次乘法和 m 次除法。

（3）“平方—乘法”算法：这是一种高效的算法。设 $m=m_k2^k+m_{k-1}2^{k-1}+\cdots+m_12+m_0$，$m_i=0,1, i=0,1,2,\cdots,k$，是指数 m 的二进制表示，则：

$$a^m=a^{m_k 2k\ +\cdots+m_1^2+m_0}$$

$$=\left(a^{m_k}\right)^{2^k}\left(a^{m_{k-1}}\right)^{2^{k-1}}\cdots\left(a^{m_1}\right)^2\left(a^{m_0}\right)$$

$$=\left(\left(\left(\left(\left(\left(a^{m_k}\right)^2 a^{m_{k-1}}\right)^2 a^{m_{k-2}}\right)^2\cdots\right)^2 a^{m_2}\right)^2 a^{m_1}\right)^2 a^{m_0}$$

此算法的流程图如图 1-3 所示。

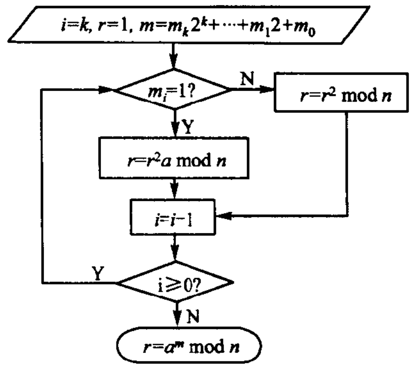


图 1-3　二进制幂模计算流程图

可见，这一算法的空间开销主要用于：存储 a,n,m，进行一次乘法，一次除法的缓存。时间开销主要用于：作 $k=\log_2 m$ 次平方，$k=\log_2 m$ 次除法，$m_0+\cdots+m_{k-1}\leqslant k$ 次乘法。也就是说，最多作 $2\log_2 m$ 次乘法和 $\log_2 m$ 次除法。这比逐次乘模算法的运算次数要少得多。

关于幂模算法，还有许多其他算法。

1.6　指数与原根

定义 1-12　设 $(a,m)=1$，使 $a^d\equiv 1 \mod m$ 成立的最小正整数称为 a 对模 m 的指数或阶，记为 $\delta_m(a)$，即：$\delta_m(a)=\min D_m(a)$，$D_m(a)=\left\{d \mid a^d\equiv 1 \mod m \quad d\in N\right\}$。

由欧拉定理知，$\varphi(m)\in D_m(a)$，所以 $\delta_m(a)$ 是存在的。

定义 1-13　如果 $\delta_m(a)=\varphi(m)$，则称 a 是模 m 的原根。

【例 1-7】　$m=10=2\times5$，$\varphi(m)=4$，则模 m 的指数表是：

a	−3	−1	1	3
$\delta_m(a)$	4	2	1	4

可见，1-3，3 模 10 的原根。

【例 1-8】　$m=15=3\times5$，$\varphi(m)=8$，则模 m 的指数表是：

a	−7	−4	−2	−1	1	2	4	7
$\delta_m(a)$	4	2	4	2	1	4	2	4

可见，模 15 没有原根。

定理 1-21　若 $a\equiv b \mod m$，$(a,m)=1$，则 $\delta_m(a)=\delta_m(b)$。

证明　设 $a^{\delta m(a)}\equiv 1 \mod m$，

由于 $a\equiv b \mod m$，所以 $b^{\delta m(a)}\equiv 1 \mod m$，

$\Rightarrow$ $\delta_m(a)\in D_m(b)$，于是 $\delta_m(a)\geqslant\delta_m(b)$，

同理可得$\delta_m(b) \geqslant \delta_m(a)$，所以$\delta_m(a)=\delta_m(b)$。定理证毕。

定理 1-22 如果$a^d \equiv 1 \mod m$，则$\delta_m(a) | d$。特别地$\delta_m(a) | \varphi(m)$。

证明 设$d=q\delta_m(a)+r$，$0 \leqslant r < \delta_m(a)$，则得：

$$
\begin{aligned}
1 &\equiv a^d \\
&\equiv a^{q\delta m(a)+r} \\
&\equiv \left(a^{\delta m(a)}\right)^q a^r \\
&\equiv a^r \mod m
\end{aligned}
$$

由于$\delta_m(a)=\min D_m(a)$，所以$r=0$。即$\delta_m(a) | d$。定理证毕。

定理 1-23 若$(a,m)=1$，$a^k \equiv a^h \mod m$，则$k \equiv h \mod \delta_m(a)$。

证明 由条件得：$a^{k-h} \equiv 1 \mod m$，所以由定理 1-22 得：$\delta_m(a) | (k-h)$，即$k \equiv h \mod \delta_m(a)$。定理证毕。

定理 1-24 若$(a,m)=1$，则$a^0,a^1,a^2,\cdots,a^{\delta_m(a)-1}$模$m$两两不同余。如果$a$是$m$的原根，则这$\varphi(m)$个数恰是模$m$的一个最小剩余系。

证明 这是定理 1-23 的直接结果。

定理 1-25 设$a^{-1}a \equiv 1 \mod m$，则$\delta_m\left(a^{-1}\right)=\delta_m(a)$。

证明 由条件得：

$\left(a^{-1}\right)^{\delta m(a)} \equiv \left(a^{-1}\right)^{\delta m(a)} \times 1 \equiv \left(a^{-1}\right)^{\delta m(a)} (a)^{\delta m(a)} \equiv \left(a^{-1}a\right)^{\delta m(a)} \equiv 1 \mod m$，

$\Rightarrow \delta_m(a) \in D_m\left(a^{-1}\right)$，得：$\delta_m(a) \geqslant \delta_m\left(a^{-1}\right)$，

同理可得$\delta_m\left(a^{-1}\right) \geqslant \delta_m(a)$，所以$\delta_m\left(a^{-1}\right)=\delta_m(a)$。定理证毕。

定理 1-26 $\delta_m(ab)=\delta_m(a)\delta_m(b)$的充要条件是$(\delta_m(a),\delta_m(b))=1$。

证明 必要性：反设：$(\delta_m(a),\delta_m(b))=k>1$，$\delta_m(a)=kh_a$，$\delta_m(b)=kh_b$，则：

$(ab)^{kh_ah_b} \equiv a^{kh_ah_b}b^{kh_ah_b} \equiv \left(a^{kh_a}\right)^{h_b}\left(b^{kh_b}\right)^{h_a} \equiv \left(a^{\delta m(a)}\right)^{h_b}\left(b^{\delta m(b)}\right)^{h_a} \equiv 1 \mod m$，所以：

$\Rightarrow kh_ah_b \in D_m(ab)$，

但$kh_ah_b < (kh_a)(kh_b) = \delta_m(a)\delta_m(b) = \delta_m(ab)$，矛盾。

充分性：$1 \equiv (ab)^{\delta m(ab)} \equiv (ab)^{\delta m(ab)\delta m(b)} \equiv (a)^{\delta m(ab)\delta m(b)} \mod m$，

$\Rightarrow \delta_m(a) | \delta_m(ab)\delta_m(b)$，由于$(\delta_m(a),\delta_m(b))=1$，

$\Rightarrow \delta_m(a) | \delta_m(ab)$；

同理可推出$\delta_m(b) | \delta_m(ab)\delta_m(a)$，$\delta_m(b) | \delta_m(ab)$，再由$(\delta_m(a),\delta_m(b))=1$，可得：

$\Rightarrow \delta_m(a)\delta_m(b) | \delta_m(ab)$。

另一方面，$(ab)^{\delta m(a)\delta m(b)} \equiv (a)^{\delta m(a)\delta m(b)}(b)^{\delta m(a)\delta m(b)} \equiv 1 \mod m$，

$\Rightarrow \delta_m(ab) | \delta_m(a)\delta_m(b)$。

总之得：$\delta_m(ab)=\delta_m(a)\delta_m(b)$。

定理 1-27 （1）$\delta_m\left(a^n\right)=\dfrac{\delta_m(a)}{(\delta_m(a),n)}$；（2）在模 m 的最小剩余系中至少有 $\varphi(\delta_m(a))$ 个数对模 m 的指数等于 $\delta_m(a)$。

证明 （1）由 $\delta_m\left(a^n\right)$ 的定义得 $1\equiv\left(a^n\right)^{\delta_m(a^n)}\equiv a^{n\delta_m(a^n)} \mod m$，所以：

$n\delta_m(a)\Big|\delta_m\left(a^n\right)$，于是：$\delta_m(a)\Big|\dfrac{n}{(\delta_m(a),n)}(\delta_m(a),n)\delta_m\left(a^n\right)$，

而 $\left(\delta_m(a),\dfrac{n}{(\delta_m(a),n)}\right)=1$，所以 $\delta_m(a)\Big|(\delta_m(a),n)\delta_m\left(a^n\right)$，从而 $\dfrac{\delta_m(a)}{(\delta_m(a),n)}\Bigg|\delta_m\left(a^n\right)$。

又由于 $\delta_m(a)\Bigg|n\dfrac{\delta_m(a)}{(\delta_m(a),n)}$，所以 $1\equiv a^{n\frac{\delta m(a)}{(\delta m(a),n)}}\equiv\left(a^n\right)^{\frac{\delta m(a)}{(\delta m(a),n)}} \mod m$，从而由定理 1-22 得 $\delta_m\left(a^n\right)\Bigg|\dfrac{\delta_m(a)}{(\delta_m(a),n)}$。

总之，$\delta_m\left(a^n\right)=\dfrac{\delta_m(a)}{(\delta_m(a),n)}$。

（2）由（1），如果 $(\delta_m(a),n)=1$，则 $\delta_m\left(a^n\right)=\delta_m(a)$，即 $\left(a^n\right)^{\delta m(a)}\equiv 1 \mod m$。满足条件 $(\delta_m(a),n)=1$ 的 n 有 $\varphi(\delta_m(a))$ 个。又因为 $(a,m)=1$，由定理 1-24，所以 $a^0,a^1,a^2,\cdots,a^{\delta m(a)-1}$ 是模 m 两两不同余，即属于不同的最小剩余类。总之，模 m 的最小剩余系中至少有 $\varphi(\delta_m(a))$ 个以 $\delta_m(a)$ 为指数。

定理 1-28 （1）如果 $n\mid m$，则 $\delta_n(a)\mid\delta_m(b)$；

（2）如果 $(m_1,m_2)=1$，则 $\delta_{m_1m_2}(a)=\left[\delta_{m_1}(a),\delta_{m_2}(a)\right]$，进一步，如果 $m=m_1\cdots m_n$，而 $m_1,m_2,\cdots,m_n$ 两两互素，则 $\delta_m(a)=\left[\delta_{m_1}(a),\delta_{m_2}(a),\cdots,\delta_{m_n}(a)\right]$，且 $\delta_m(a)\mid\left[\varphi(m_1),\varphi(m_2),\cdots,\varphi(m_n)\right]$。

证明 （1）根据 $a^{\delta_m(a)}\equiv 1 \mod m$ 和 $n\mid m$ 可得到 $a^{\delta_m(a)}\equiv 1 \mod n$，所以 $\delta_n(a)\mid\delta_m(b)$；（2）由于 $m_1\big|(m_1m_2),m_2\big|(m_1m_2)$，由（1）得 $\delta_{m_1}(a)\big|\delta_{m_1m_2}(a),\delta_{m_2}(a)\big|\delta_{m_1m_2}(a)$，所以 $\left[\delta_{m_1}(a),\delta_{m_2}(a)\right]\big|\delta_{m_1m_2}(a)$。

另一方面，显然，$a^{\left[\delta_{m_1}(a),\delta_{m_2}(a)\right]}\equiv 1 \mod m_1$，$a^{\left[\delta_{m_1}(a),\delta_{m_2}(a)\right]}\equiv 1 \mod m_2$，再由于 $(m_1,m_2)=1$，所以 $a^{\left[\delta_{m_1}(a),\delta_{m_2}(a)\right]}\equiv 1 \mod m_1m_2$，即 $\delta_{m_1m_2}(a)\mid\left[\delta_{m_1}(a),\delta_{m_2}(a)\right]$。

总之 $\delta_{m_1m_2}(a)=\left[\delta_{m_1}(a),\delta_{m_2}(a)\right]$，同理可证 $m_1,m_2,\cdots,m_n$ 两两互素时 $\delta_{m_1m_2\cdots m_n}(a)=\left[\delta_{m_1}(a),\delta_{m_2}(a),\cdots,\delta_{m_n}(a)\right]$。

由于 $\delta_{m_1}(a)\big|\varphi(m_1),\delta_{m_2}(a)\big|\varphi(m_2),\cdots,\delta_{m_n}(a)\big|\varphi(m_n)$，所以 $\delta_{m_i}(a)\big|[\varphi(m_1),\varphi(m_2),\cdots,\varphi(m_n)]$，$i=1,2,\cdots,n$，于是 $[\delta_{m_1}(a),\delta_{m_2}(a),\cdots,\delta_{m_n}(a)]\mid[\varphi(m_1),\varphi(m_2),\cdots,\varphi(m_n)]$。所以 $\delta_m(a)\mid[\varphi(m_1),\varphi(m_2),\cdots,\varphi(m_n)]$。

定理 1-29 $l\geqslant 3$ 时，$\delta_{2^l}(a)\big|2^{l-2}$。

证明 因为 $(a,2^l)=1$，所以可设 $a=2k+1$，对 $l\geqslant 3$ 应用归纳法。

$l=3$ 时，$a^{2^{l-2}}=a^2=4k(k+1)+1\equiv 1 \mod 2^3$，

设 $l=n$ 时 $a^{2^{l-2}}\equiv 1 \mod 2^l$，即 $2^n\Big|\left(a^{2^{n-2}}-1\right)$，

$l=n+1$ 时，$a^{2^{l-2}}-1=a^{2^{n-1}}-1=\left(a^{2^{n-2}}-1\right)\left(a^{2^{n-2}}+1\right)\left(a^{2^{n-2}}-1\right)\left(a^{2^{n-2}}-1+2\right)$，

由归纳得：$2^n\Big|\left(a^{2^{n-2}}-1\right)$，$2\Big|\left(a^{2^{n-2}}-1+2\right)$，所以 $2^{n+1}\Big|\left(a^{2^{n-1}}-1\right)$，即 $a^{2^{l-2}}\equiv 1 \mod 2^l$。定理证毕。

定理 1-30 m 存在原根的充要条件是：$m=1,2,4,p^\alpha,2p^\alpha$，$p$ 是奇素数。

证明 必要性，如果 $m\neq 1,2,4,p^\alpha,2p^\alpha$，则有

（1）$m=2^{\alpha_0}p_1^{\alpha_1}\cdots p_n^{\alpha_n}$，$\alpha_0\geqslant 3,n=0$，或

（2）$m=2^{\alpha_0}p_1^{\alpha_1}\cdots p_n^{\alpha_n}$，$\alpha_0\geqslant 2,n\geqslant 1$，或

（3）$m=2^{\alpha_0}p_1^{\alpha_1}\cdots p_n^{\alpha_n}$，$\alpha_0\geqslant 0,n\geqslant 2$。

因为：$\varphi\left(2^{\alpha_0}\right)=\begin{cases}1, & \alpha_0=0,1\\ 2, & \alpha_0=2\\ 2^{\alpha_0-2}, & \alpha_0\geqslant 3\end{cases}$，$\left[\varphi\left(p_2^{\alpha_2}\right),\cdots,\varphi\left(p_n^{\alpha_n}\right)\right]\geqslant 2$，

所以由定理 1-28 得：$\delta_m(a)\leqslant\left[\varphi\left(2^{\alpha_0}\right),\varphi\left(p_2^{\alpha_2}\right),\cdots,\varphi\left(p_n^{\alpha_n}\right)\right]<\varphi\left(2^{\alpha_0}\right)\varphi\left(p_2^{\alpha_2}\right)\cdots\varphi\left(p_n^{\alpha_n}\right)=\varphi(m)$。所以，$m$ 没有原根。

充分性：可验证 $m=1,2,4$ 有原根 $1,1,-1$。对 $m=p^\alpha,2p^\alpha$ 时，证明较复杂，略去。

定理证毕。

1.7 习题

1．证明：对任意正整数 n，有：

（1）$6\mid n(n+1)(n+2)$；

（2）$24\mid n(n+1)(n+2)(n+3)$；

（3）若 $2\nmid n$，则 $8\mid n^2-1$，$24\mid n(n^2-1)$；

（4）若 $2\nmid n$，$3\nmid n$，则 $24\mid n^2+23$；

（5）$6\mid n^3-n$；

（6）$30\mid n^5-n$；

（7）$42 \mid n^7 - n$。

2．证明：对任意正整数 d，有：

（1）$13 \nmid 3^d, 3^d+1, 3^d \pm 2, 3^d+3, 3^d-4, 3^d \pm 5, 3^d \pm 6$；

（2）$13 \nmid 4^d, 4^d \pm 2, 4^d \pm 5, 4^d \pm 6$。

3．证明对任意 $n \geqslant 1$，$(n!+1, (n+1)!+1)=1$。

4．求 $[198, 252]$，$[482, 1687]$。

5．证明：$(a,b)[a,b]=ab$。

6．设 $(a,b)=1$，证明 $(d,ab)=(d,a)(d,b)$。

7．求满足 $(a,b)=10$，$[a,b]=100$ 的全体正整数对 a,b。

8．设 $m>n$，证明 $2^n-1 \mid 2^m-1$ 的充要条件是 $n \mid m$。

9．设 p 是正整数 n 的最小素因数，证明：如果 $p > \sqrt[3]{n}$，则 $\frac{n}{p}$ 是素数。

10．设 $p_1 \leqslant p_2 \leqslant p_3$ 是素数，n 是正整数，若 $p_1 p_2 p_3 \mid n$，则 $p_1 \leqslant \sqrt[3]{n}$，$p_2 \leqslant \sqrt{\frac{n}{2}}$。

11．证明 $4k-1$，$6k-1$ 形式的素数有无穷多个。

12．证明 $a+b \equiv a+(b \bmod m) \bmod m \equiv (a \bmod m)+(b \bmod m) \bmod m$。

13．证明 $ab \equiv a(b \bmod m) \bmod m \equiv (a \bmod m)(b \bmod m) \bmod m$。

14．计算下列 Legendre 符号的值：

$\left(\frac{13}{47}\right)$，$\left(\frac{71}{73}\right)$，$\left(\frac{102}{81}\right)$，$\left(\frac{-35}{97}\right)$，$\left(\frac{77}{103}\right)$，$\left(\frac{-23}{131}\right)$，$\left(\frac{7}{223}\right)$，$\left(\frac{163}{257}\right)$，$\left(\frac{91}{563}\right)$，$\left(\frac{254}{773}\right)$，$\left(\frac{369}{1949}\right)$，$\left(\frac{1215}{4523}\right)$。

15．判断以下各数是否为平方剩余：

（1）1，2，4，5，7，9，10，11，13 模 15；

（2）2，4，7，10，11，19，20 模 23。

16．求下列各数的全部原根：

7，13，14，17，19。

1.8 实验题

1．编写求两个正整数最大公因子，模逆元的的计算机程序。

2．编写实现大数四则运算（除法要求商、余数）的计算机程序，并测试运算时间。

3．编写计算大数的“幂-模”运算：$a^m \bmod n$ 的计算机程序，并测试计算时间。

4．应用公式 $e \approx 1+\frac{1}{1!}+\frac{1}{2!}+\cdots+\frac{1}{n!}$，编写计算自然对数的底 e 到 1000 位的程序，并测试运算时间。

5．编写概率算法的素性检测算法，求取 100 位十进制的素数，并测试时间。

第 2 章　近世代数基础

2.1　群论初步

定义 2-1　设 G 是一个非空集合，其上存在一种运算，称为乘法"$\circ$"（书写时常省略），如果下列公理满足，则称 G 是一个群，记为 $(G, \circ)$ 或简记为 G：

（1）结合律：如果 a，b，$c \in G$，则 $(ab)c = a(bc)$；

（2）单位元：存在一个元 $e \in G$，使得 $ea = a$，$ae = a$ 对于一切 $a \in G$ 成立，元素 e 称为的 G 单位元，简记为 1；

（3）逆元：对于每一个元 $a \in G$，存在元 $\bar{a} \in G$，使得 $\bar{a}a = e$，$a\bar{a} = e$，元素 $\bar{a}$ 称为 a 的逆元，记为 a^{-1}。

进一步，如果对于 $\forall a,b \in G$，$ab = ba$，则称 G 是一个可换群或 Abel 群。对可换群，运算通常称作加法"+"，记为 $(G, +)$ 或简记为 G，其单位元称为零元，记为 0，a 的逆元称为 a 的负元，记为 $-a$。

【例 2-1】　设 $G = \{0,1\}$，在其上定义运算：$0 \oplus 0 = 0, 1 \oplus 0 = 0 \oplus 1 = 1, 1 \oplus 1 = 0$，可验证，这是一个 Abel 群，单位元是 0，0 的负元是 0，1 的负元是 1。

【例 2-2】　设 $n > 1$ 是一个正整数，$Z_n = \{0,1,2,\cdots,n-1\}$ 是模 n 的完全剩余类，在 Z_n 上定义一个运算称为模 n 的加法 $\oplus_n$，即：对 $\forall a,b \in Z_n$，$a \oplus_n b = a + b \bmod n$，则 $(Z_n, \oplus_n)$ 是一个 Abel 群；

如果 n 是一个素数 p，$Z_p^* = \{1,2,\cdots,p-1\}$，定义：$\forall a,b \in Z_p$，$a \otimes_p b = ab \bmod p$，则 $(Z_p^*, \otimes_p)$ 也是一个 Abel 群。

证明　（1）对 $\forall a,b,c \in Z_n$，

$$
\begin{aligned}
(a \oplus_n b) \oplus_n c &= (a + b \mod n) + c \mod n \\
&= a + b + c \mod n \\
&= a + (b + c \mod n) \mod n \\
&= a \oplus_n (b \oplus_n c),
\end{aligned}
$$

所以结合律成立；

对 $\forall a \in Z_n$，$0 \oplus_n a = 0 + a \mod n = a + 0 \mod n = a \oplus_n 0$，所以 0 是零元；

对 $\forall a \in Z_n$，$(n-a) \oplus_n a = n - a + a \mod n = 0 \mod n = 0$，所以，$a$ 的零元是 $n-a$；

对 $\forall a,b \in Z_n$，$a \oplus_n b = a + b \mod n = b + a \mod n = b \oplus_n a$。所以，$\oplus_n$ 是可换的。

总之，$(Z_n, \oplus_n)$ 是一个可换群。

（2）对 $\forall a,b,c \in Z_n^*$，

$$
\begin{aligned}
(a\otimes_p b)\otimes_p c &= (ab \mod p)c \mod p \\
&= abc \mod p \\
&= a(bc \mod p) \mod p \\
&= a\oplus_p (b\oplus_p c),
\end{aligned}
$$

所以结合律成立；

对$\forall a\in Z_p^*$，$1\otimes_p a=1\cdot a \mod p=a\cdot 1 \mod p=a\otimes_p 1$，所以 1 是单位元；

对$\forall a\in Z_p^*$，由于p是素数，所以$(a,p)=1$，由定理 1-8 知，存在$b\in Z_p^*$，$ab\equiv 1 \mod p$，即$a\otimes_p b=ab \mod p=1$所以，a有逆元b；

对$\forall a,b\in Z_p^*$，$a\otimes_p b=ab \mod p=ba \mod p=b\otimes_p a$。所以，$\otimes_p$是可换的。

总之，$(Z_p^*,\otimes_p)$是一个可换群。定理证毕。

【例 2-3】 设$GL(R,n)$是全体n阶非奇异方阵的集合，在其上定义乘法为普通的矩阵乘法，则$GL(R,n)$就是一个群，是非 Abel 群。

定义 2-2 设G是一个群，$\varnothing\neq S\subset G$，如果关于$G$中的运算$S$也是一个群，则称$S$是$G$的一个子群。

定理 2-1 设G是一个群，$\varnothing\neq S\subset G$，则$S$是$G$的子群的充要条件是：$S$关于$G$的运算是封闭的，即：对于$\forall a,b\in S\subset G$，$ab\in S$，$a^{-1}\in S$。

证明：必要性是自然的。

充分性：由于S的乘法是从G中继承而来的，所以结合律是自然成立的。

对于任意$a\in S$，由逆元的封闭性得$a^{-1}\in S$，再由乘法的封闭性得：$e=aa^{-1}\in S$，即G的单位元也含在S中，显然，它也是S的单位元。

确定了单位元后，逆元的存在性就是条件给出的。所以S是一个群。定理证毕。

定义 2-3 如果群$G=\{a^0=e,a^{\pm1},a^{\pm2},\cdots\}$，则称$G$是一个循环群，记为$\langle a\rangle$。

定理 2-2 设G是一个群，则对$\forall a\in G$，$\langle a\rangle=\{a^0=e,a^{\pm1},a^{\pm2},\cdots\}$是$G$的一个子群，称为由元$a$生成的循环群。

【例 2-4】 设Q是除 0 外的一切有理数，运算就是普通的乘法，则Q是一个 Abel 群，它有下列子群：

1）正有理数之集；

2）任一元的全体整数幂$\{a^0,a^{\pm1},a^{\pm2},\cdots\}$；

3）集合$\{-1,+1\}$；

4）集合$\{1\}$。

记集合A中元素的个数为$|A|$或$\#(A)$。

定义 2-4 如果群G只含有限个元素，则称G是一个有限群，G中元素的个数称为G群的阶。

定义 2-5 设G是一个群，则$\#(\langle a\rangle)$ 称为元a的阶或周期。

定理 2-3 设G是一个有限群，S是G的一个子群，则$\#(S)|\#(G)$。

证明：设 $aS=\{as|s\in S\}$，称为 S 的陪集，则：

（1）$|aS|=|S|$；（易证）

（2）对于 $\forall a,b\in G$，则 $aS=bS$，或 $aS\cap bS=\varnothing$。

证明如下：

对于 $\forall s\in S$，$sS\subset S$；

反之，对于 $\forall s\in S$，$s=se\in sS,\Rightarrow S\subset sS$，所以：$sS=S$。

如果 $aS\cap bS\neq\varnothing$，则存在 $s_1,s_2\in S$，$as_1=bs_2$，

$\Rightarrow b=a(s_1s_2^{-1})$，$s_1s_2^{-1}\in S$，所以：$bS=a(s_1s_2^{-1})S=a\left[(s_1s_2^{-1})S\right]=aS$，这样断言就得到了证实。

由（1）、（2）得知，G 是一些互不相交的陪集的并集，故有：$|G|=k|S|$。

即：$\#(S)|\#(G)$。定理证毕。

推论 1：有限群的元素的阶必为群的阶的因子。

推论 2：如果群的阶是一个素数，则必为循环群。

2.2 域论初步

定义 2-6 设 F 是一个非空集合，F 中定义了两种运算，一种称为“加法”，记为“+”；另一种称为“乘法”，记为“$\circ$”，它们满足下面的公理：

（1）$(F,+)$ 是一个 Abel 群，单位元称为零元，通常记为 0；

（2）$F^*=F\setminus\{0\}$, $(F^*,\circ)$ 是一个 Abel 群，单位元通常记为 1 或 e；

（3）分配律：对于 $\forall a,b,c\in F$，$a(b+c)=ab+ac$。

则称 $(F,+,\circ)$ 是一个域（Field）。具有有限个元素的域称为有限域，或 Galois 域。

定义 2-7 如果 F 是一个域，如果存在正整数 n，使对 $\forall a\in F$，$na=0$，则 n 中最小者称为域 F 的特征，记为 $Char(F)$。如果这种正整数 n 不存在，则称域 F 的特征为 0。

定义 2-8 如果 $(E,+,\circ)$ 是一个域，$F\subset E$ 是 E 的子集，如果 $(F,+,\circ)$ 也是一个域，则称 F 是 E 的子域，E 是 F 的扩域。

【例 2-5】 设 p 是一个素数，$Z_p=\{0,1,2,\cdots,p-1\}$ 是模 p 的剩余类集，在其上定义：

$a\oplus_p b=(a+b)\bmod p$，$a\otimes_p b=(ab)\bmod p$，

则 $(Z_p,\oplus_p,\otimes_p)$ 是一个有限域，且其特征为 p，常记为：$GF(p)$，或 F_p，Z_p。

证明 由例 2-2 知 $(Z_p,\oplus_p)$ 是一个 Abel 群，$(Z_p^*,\otimes_p)$ 也是一个 Abel 群。所以只须验证分配律即可。

$$\begin{aligned}(a\oplus_p b)\otimes_p c&=(((a+b)\bmod p)c)\bmod p\\&=(ac+bc)\bmod p=(ac\bmod p+bc\bmod p)\bmod p\\&=(a\otimes_p c)\oplus_p(b\otimes_p c)。\end{aligned}$$

【例 2-6】 全体有理数之集 Q，实数集 R 在普通加法、乘法下构成一个域，分别称为有

理数域、实数域。它们的特征都是 0。

定义 2-9 设 $(F, +, \circ)$ 是一个域，则

$$f(x)=a_0x^0+a_1x^1+\cdots+a_nx^n \quad (a_i\in F, 0\leqslant i\leqslant n,\ a_n\neq 0)$$

称为域 F 上 x 的多项式。常用 $F[x]$ 表示域 F 上 x 全体多项式的集合。

定义 2-10 若多项式不能在定义它的域上进行因式分解，则称该多项式为不可约多项式，或既约多项式。

例如：x^2+1，x^2+x+1，是有理数域或实数域上的不可约多项式；x^2+x+1，x^4+x+1，$x^8+x^4+x^3+x+1$ 是 Z_2 上的不可约多项式。

但是，在 Z_2 上 $x^2+1=(x+i)(x+i)$，所以不是既约的。

验证一个多项式是否在相应域中可约也是一件困难的事，类似于素性检验。我们可以用多项式次数二分之一及其以下的所有多项式（不可约多项式）试除来确定不可约性。一次因式可以用域中元素代入多项式试根来确定。如：任意有理数或实数都不是 x^2+1 的根，所以 x^2+1 没有一次因式，从而在有理数域或实数域上不可约。

$Z_2=\{0,1\}$，而 0，1 都不是 x^2+x+1 的根，从而 x^2+x+1 没有一次因式，所以 x^2+x+1 是 Z_2 上的不可约多项式。

类似地，可以验证 x^4+x+1 也没有一次多项式。但是，这还不够，我们还需要验证其没有二次因式。二次不可约多项式只有一个 x^2+x+1，用其除 x^4+x+1，即可确定不可约性：

$$\begin{array}{r|l}
 & x^2+x \\
\hline
x^2+x+1 & x^4+0x^3+0x^2+x+1 \\
 & x^4+x^3+x^2 \\
\hline
 & \quad x^3+x^2+x \\
 & \quad x^3+x^2+x \\
\hline
 & \qquad\qquad 1
\end{array}$$

注：二次因式还有 x^2，$x^2+1=(x+1)(x+1)$，$x^2+x=x(x+1)$，它们都是一次因式 x，$x+1$ 的乘积，因已经检验了一次因式的不可约性，从而不必再用它们试除了。

类似地可验证 $x^8+x^4+x^3+x+1$ 的不可约性：即用：$x, x+1$，x^2+x+1，x^3+x+1, x^3+x^2+1，$x^4+x+1, x^4+x^3+x^2+x+1$，x^4+x^3+1 试除。

后面我们可以看到，不可约多项式是我们构造有限域的基础，所以，其重要性不言而喻。但这已超出了本教材的范围，有兴趣的读者可参看有关有限域的著作。

【例 2-7】 设 F 是一个域，$f(x)$ 是 $F[x]$ 中的一个 n 次不可约多项式，令：

$$F[x]/(f(x))=\left\{a_0+a_1x+\cdots+a_{n-1}x^{n-1}\,\middle|\,a_i\in F, 0\leqslant i\leqslant n-1\right\}$$

表示系数在 F 上的多项式对 $f(x)$ 取模的剩余类的集合。

在其上定义加法运算 $\oplus$：

$$a(x)\oplus b(x)=(a(x)+b(x))\bmod(f(x))；$$

定义乘法运算 $\otimes$：

$$a(x)\otimes b(x)=(a(x)b(x))\bmod(f(x))。$$

则$\left({F[x]}/{(f(x))}, \oplus, \otimes \right)$是一个有限域。

如果$F=Z_p$。则${F[x]}/{(f(x))}={Z_p[x]}/{(f(x))}$有$p^n$个元素，通常记为：$GF(p^n)$。

定义 2-11 设h：$E_1 \to E_2$是域E_1，E_2间的一一对应，对任意的$a,b\in E_1$，有：$h(a+b)=h(a)+h(b)$，$h(ab)=h(a)h(b)$，则称h是域E_1，E_2间的一个同构映射，域E_1，E_2称为是同构的，记为：$E_1 \overset{h}{\cong} E_2$。

同构的域具有完全相同的域的性质，域论中，将同构的域视为相同而不加区别（惟一的差别仅仅是元素的表示形式，运算的记号不同而已）。

【例 2-8】 设$F=R$是实数域，$f(x)=x^2+1$，则$\left({R[x]}/{(f(x))}, \oplus, \otimes \right)$与复数域$C$同构。

解 ${R[x]}/{(x^2+1)}=\{a+bx \mid a,b\in R\}$。

$$(a+bx)+(c+dx)=(a+c)+(b+d)x,$$

$$\begin{aligned}(a+bx)\ (c+dx)&=(ac)+(ad+bc)x+(bd)x^2=(ac)+(ad+bc)x+(bd)(x^2+1-1)\\ &=(ac-bd)+(ad+bc)x\end{aligned}$$

定义π：${R[x]}/{(x^2+1)} \to C=\{a+bi \mid a,b\in R\}$，$\pi$：$a+bx \mapsto a+bi$。

$$\begin{aligned}\pi((a+bx)+(c+dx))&=\pi((a+c)+(b+d)x)=(a+c)+(b+d)i\\ &=(a+bi)+(c+di)=\pi(a+bx)+\pi(c+dx)\end{aligned}$$

$$\begin{aligned}\pi((a+bx)\cdot(c+dx))&=\pi((ac-bd)+(ad+bc)x)=(ac-bd)+(ad+bc)i\\ &=(a+bi)(c+di)=\pi(a+bx)\pi(c+dx)\end{aligned}$$

$$0=\pi(a+bx)=a+bi \Leftrightarrow a=0,b=0 \Leftrightarrow a+bx=0$$

所以，π是同构，即：${R[x]}/{(x^2+1)} \overset{\pi}{\cong} C$。

定义 2-12 除零子域$\{0\}$外没有其他真子域的域称为素域。

可以证明，有理数域Q和模p剩余类Z_p都是素域，其特征分别为 0，p。

定理 2-4 域的特征或者是 0，或者是素数p。

证明 设域F的特征不是 0，为n。反设$n=pq$，$1<p,q<n$。e是F^*的单位元，则$0=ne=(pq)e=(pe)(qe)$，得到$pe=0$或者$qe=0$。不妨设$pe=0$，由此可得对$\forall a\in F, pa=p(ea)=(pe)a=0$，于是，$F$的特征不大于$p$，从而小于$n$，此与$F$的特征是$n$矛盾，所以$n$是素数。定理证毕。

定理 2-5 （1）特征为 0 的域的素子域同构于有理数域Q；（2）特征为素数p的域的素子域同构于Z_p。

证明 （1）设F的特征为 0，P为F的素子域。

第一步要证明：$P=\{(me)(ne)^{-1} \mid m,n \in Z\}$。

a）首先因为 P 是 F 的子域，所以可以证明单位元 $e\in P$，进一步，

$\Rightarrow$ 对$\forall$整数 m，$n\in Z$，$n\neq 0$，$me\in P$，$(ne)^{-1}\in P$，$\Rightarrow (me)(ne)^{-1}\in P$。

所以，$P\supset\{(me)(ne)^{-1} \mid m,n \in Z\}$；

b）其次，再证$\{(me)(ne)^{-1} \mid m,n \in Z\}$是一个域。

易见$(n_1n_2e)^{-1}=(n_1e)^{-1}(n_2e)^{-1}$，由此不难验证：

$(m_1e)(n_1e)^{-1}+(m_2e)(n_2e)^{-1}=(m_1n_2+m_2n_1e)(n_1n_2e)^{-1}$；

$(m_1e)(n_1e)^{-1}(m_2e)(n_2e)^{-1}=(m_1m_2e)(n_1n_2e)^{-1}$；

$(m_1e)(n_1e)^{-1}(m_2e)(n_2e)^{-1}=(m_1m_2e)(n_1n_2e)^{-1}$；

$-(me)(ne)^{-1}=((-m)e)(ne)^{-1}$；

所以$\{(me)(ne)^{-1} \mid m,n \in Z\}$是 F 的一个子域。

由于 P 无非 0 真子域，所以 $P=\{(me)(ne)^{-1} \mid m,n \in Z\}$。

第二步证明 P 与有理数域 Q 同构。

定义π ：$P\to Q$ ，$(me)(ne)^{-1}\mapsto\dfrac{m}{n}$；

a）易见π是满射。

b）$\pi\left((m_1e)(n_1e)^{-1}+(m_2e)(n_2e)^{-1}\right)=\pi\left((m_1n_2+m_2n_1)e\right)(n_1n_2e)^{-1}=\dfrac{m_1n_2+m_2n_1}{n_1n_2}$

$$=\frac{m_1}{n_1}+\frac{m_2}{n_2}=\pi\left((m_1e)(n_1e)^{-1}\right)+\pi\left((m_2e)(n_2e)^{-1}\right)$$

$$\pi\left((m_1e)(n_1e)^{-1}\cdot(m_2e)(n_2e)^{-1}\right)=\pi\left(m_1m_2e\right)(n_1n_2e)^{-1}=\frac{m_1m_2}{n_1n_2}$$

$$=\frac{m_1}{n_1}+\frac{m_2}{n_2}=\pi\left((m_1e)(n_1e)^{-1}\right)\pi\left((m_2e)(n_2e)^{-1}\right)$$

所以π是保运算的。

c）设$\pi\left((m_1e)(n_1e)^{-1}\right)=\pi\left((m_2e)(n_2e)^{-1}\right)$，$\Rightarrow\dfrac{m_1}{n_1}=\dfrac{m_2}{n_2}$，

令$\dfrac{m_2}{m_1}=\dfrac{n_2}{n_1}=k$，由等式$(n_1n_2e)^{-1}=(n_1e)^{-1}(n_2e)^{-1}$及可换性：

$(m_2e)(n_2e)^{-1}=(km_1e)(kn_1e)^{-1}=(ke)(m_1e)(ke)^{-1}(n_1e)^{-1}=(m_1e)(n_1e)^{-1}$，所以$\pi$是单射。

总之：$P\overset{\pi}{\cong}Q$。

（2）设 F 的特征是素数 p，P 是 F 的素子域，则 P 必含 $0,1e,2e,\cdots,(p-1)e$。

第一步证明：$P=\{0,\ 1e,\ 2e,\ \cdots,\ (p-1)e\}$。

设 $0\leqslant m,n<p$，则：

$$(me)+(ne)=(m+n)e=\left[\frac{m+n}{p}\right]pe+(m+n \mod p)e=(m+n \mod p)e\in P,$$

$$(me)(ne)=(\underbrace{e+\cdots+e}_{m个})(ne)=\underbrace{e(ne)+\cdots+e(ne)}_{m个}=\underbrace{(ne)+\cdots+(ne)}_{m个}=(mne)$$

$$=\left[\frac{mn}{p}\right]e+(mn \bmod p)e=(mn \bmod p)e\in P$$

$$(me)(ne)=(e+\cdots+e)(ne)=e(ne)+\cdots+e(ne)=(ne)+\cdots+(ne)$$

$$=\underbrace{e+\cdots+e}_{n个}+\cdots+\underbrace{e+\cdots+e}_{n个}=(mne)$$

$$=\left[\frac{mn}{p}\right]pe+(mn \bmod p)e=(mn \bmod p)e\in P。$$

所以，$\{0,1e,2e,\ldots,(p\text{-}1)e\}$是一个域。由于$P$没有非0真子域，所以$P=\{0,1e,2e,\ldots,(p\text{-}1)e\}$。

定义自然映射π ：$P\to Z_p,(me)\mapsto m$。

类似于（1）可以验证π一个同构，即：$P\overset{\pi}{\cong}Z_p$。定理证毕。

定理 2-6 有限域的阶必是p^n，其中p是素数，是域的特征。

证明 设F的阶为q，P_1是其素子域，如果$F=P_1$，那么$\#(F)=p^1$，结论成立。

如果$F\neq P_1$，则有$e_2\in F\setminus P_1$，令$P_2=\{a_1+a_2e_2 \mid a_1,a_2\in P_1\}$，如果$F=P_2$，那么$\#(F)=\#(P_2)=p^2$，结论成立，否则，继续下去，得到$e_2\in P_2,e_3\in P_3,\cdots,e_m\in P_m$，$\#(P_m)=p^m$。由于$\#(F)$有限，此步骤必终止于有限步$n$，即：必有$n$，$F=P_n$，此时$\#(P_n)=p^n$。定理证毕。

定理 2-7 有限域的乘法群必是循环群。

证明 设n是F^*中元的最大阶，则$n\leqslant q-1$，$q=\#(F)$。

设a的阶为n，b的阶为$m<n$，p是素数，令$n=p^s n',m=p^t m'$，$(p,n')=1,(p,m')=1$。

如果$s<t$，则可以验证$a^{p^S}b^{m'}$的阶为$p^t n'>n$，此与n是元素的最大阶矛盾，所以必有$s\geqslant t$。对n的任何素因子都如此讨论，则知$m \mid n$。

所以，对$\forall b\in F,b^n=e$。即F中任意元素都是多项式$x^n-1=0$的根，0 显然不是其根，于是$q-1\leqslant n$。

总之，$n=q-1$，于是$\#\{e=a^0,a^1,a^2,\cdots,a^{n-1}\}=n=q-1=\#(F)$，所以$F^*=\{e,a,a^2,\cdots,a^{q-2}\}=\langle a\rangle$，即$F^*$是循环群。定理证毕。

循环群F^*的生成元a称为域F的本原元。易见本原元不惟一，本原元的个数为$\varphi(q-1)$。

本原元是有限域密码技术应用中的一个重要对象，特别在有限域的表示中具有重要应用，然而，求本原元是困难的。

定义 2-13 设$f(x)$是定义在域F上的一个多项式，即：$f(x)\in F[x]$，E为域F的一个扩域，如果满足：

（1）$f(x)$在$E[x]$中可以完全分解成一次多项式（因子）的乘积；

（2）在介于E，F间的任何域中，$f(x)$不能完全分解。

则称E是$f(x)$的分裂域。

定义 2-14 设 E 是域，如果任意非常数多项式 $f(x) \in E[x]$ 都可以在 E 中完全分解，则称 E 是一个代数闭域，或是代数封闭的。

例如：复数域是代数闭域，而有理数域和实数域都不是代数封闭的。

定理 2-8 对于任意素数 p 及正整数 n，一定存在一个具有 p^n 个元素的有限域 $GF(p^n)$。在同构的意义下，它就是 Z_p 上多项式 $f(x) = x^{p^n} - x$ 的分裂域。

存在性由例 2-7 给出。唯一性这里不证，有兴趣的读者可参看有关著作。

定义 2-15 设 E 是 F 的一个扩域，$\alpha \in E$，如果存在多项式 $f(x) \in F[x]$，$f(\alpha) = 0$。则称 α 为 F 上的一个代数元，满足 $f(\alpha) = 0$ 的，最低次数的，首项系数为 1 的多项式称为 α 的极小多项式；反之，如果对 $\forall f(x) \in F[x]$，$f(\alpha) \neq 0$，则称 α 为 F 上的一个超越元。

例如：设 $F = Q$ 是有理数域，$E = R$ 是实数域，$\sqrt{2} \in R$，对 $f(x) = x^2 - 2 \in Q[x]$，有：$f(\sqrt{2}) = 0$，所以，$\sqrt{2}$ 是 Q 上的代数元。但是 $\pi \in R$ 却不是 $Q[x]$ 中任意多项式的根，从而它是 Q 上的超越元。

定义 2-16 设域 E 是域 F 的一个扩域，S 是 E 的任一子集，如果记 L 为包含 F,S 的，E 的全体子域，即：$L = \{G \mid E \supset G \supset (F \cup S), G\text{是}E\text{的子域}\}$，则：

$F(S) = \bigcap\limits_{G \in L} G$ 也是 E 的一个子域。事实上，它是包含 F,S 的最小子域，称为 F 上由 S 生成的 E 的子域，或称为添加 S 到 F 所成的 F 的扩（张）域。

如果 $S = \{\alpha\}$ 是一个元素的集合，则称这个扩张是单纯扩张，记：$F(S) = F(\alpha)$，α 称为 $F(\alpha)$ 的本原元。如果 α 是 F 上的代数元则称 $F(\alpha)$ 是单纯代数扩张。

密码学中，最常用的是有限域，如何构造一个有限域呢？下面的定理将回答这个问题，并给出了具体构造的方法。

定理 2-9 设域 E 是域 F 的一个扩域，

（1）$\alpha \in E$ 是 $F[x]$ 中 n 次不可约多项式 $f(x)$ 的根（称为 F 上 n 次代数元），则 $F(\alpha)$ 由 $F[x]$ 中次数小于 n 的多项式组成，加法运算就是普通的多项式加法，乘法运算就是普通的多项式乘法模 $f(x)$。即：$F(\alpha) \cong F[x]/(f(x))$。$F(\alpha)$ 也称为 F 的 n 次代数扩张。

（2）$\alpha \in E$ 是 F 上的超越元，则 $F(\alpha)$ 由系数属于 F 的多项式的商组成，其加法，乘法就是普通的有理函数的加法，乘法。即：$F(\alpha) \cong \left\{ \left. \dfrac{p(\alpha)}{q(\alpha)} \right| p(\alpha), q(\alpha) \in F[\alpha] \right\}$。

从例 2-7 可见，如果 $F = Z_p$，p 是素数，α 是 n 次代数元，则：$F(\alpha) = \{a_0 + a_1\alpha + \cdots + a_{n-1}\alpha^{n-1} \mid a_i \in Z_p, 0 \leqslant i \leqslant n-1\}$，可见，$F(\alpha)$ 的阶是 p^n，即 $F(\alpha) = GF(p^n)$。

超越扩张不构造有限域，本书也就不作过多关注了。其实（1）的方法不是别的，就是例 2-7 给出的方法。

另一方面，从表达式 $F(\alpha) = \{a_0 + a_1\alpha + \cdots + a_{n-1}\alpha^{n-1} \mid a_i \in Z_p, 0 \leqslant i \leqslant n-1\}$ 可见，$F(\alpha)$ 是 F 的 n 维线性空间，$1, \alpha, \alpha^2, \cdots, \alpha^{n-1}$ 是一组基。线性空间及基方法在有限域的快速计算中具有重要的作用。

2.3 有限域中的计算

从以上定理可以看出，任何有限域$GF(p^n)$都可以通过Z_p的n次代数扩张得到。下面以具体的例子来详细描述这一扩张过程以及域中的运算。

【例 2-9】 设基域是$Z_5=\{0,1,2,3,4\}$，构造$GF(5^2)$。

解 选首选择一个Z_5上的2次不可约多项式：$f(x)=x^2-2$。（对$\forall x\in Z_5$，$f(x)\neq 0$）。

记$f(x)=x^2-2$的一个根为α（也可按照习惯记为$\sqrt{2}$），则$\alpha^2=2$，

$GF(5^2)=\{a_0+a_1\alpha \mid a_0,a_1\in Z_5\}$。

选定基：$1,\alpha$后，可表示为：

$GF(5^2)=\{(a_0,a_1)\mid a_0,a_1\in Z_5\}$。

加法，数乘运算可描述为：

（1）$(a_0,a_1)+(b_0,b_1)=(a_0+b_0,a_1+b_1)$；（2）$k(a_0,a_1)=(ka_0,ka_1)$。

乘、除法运算就不是这么简单了，下面来推导乘、除法公式：

$$(a_0+a_1\alpha)(b_0+b_1\alpha)=a_0b_0+(a_1b_0+a_0b_1)\alpha+a_1b_1\alpha^2$$
$$=(a_0b_0+2a_1b_1)+(a_1b_0+a_0b_1)\alpha\text{，（因为}\alpha^2=2\text{）}$$

所以有：

（3）$(a_0,a_1)(b_0,b_1)==(a_0b_0+2a_1b_1,a_1b_0+a_0b_1)$。

特别地，$(a_0,a_1)^2=(a_0^2+2a_1^2,2a_0a_1)$。

设$0\neq a_0+a_1\alpha\in GF(5^2)$，则$\gcd(a_0+a_1x,x^2-2)=1$。根据辗转相除法，存在$Z_5$中多项式$p(x),q(x)$，使得$p(x)(a_0+a_1x)+q(x)(x^2-2)=1$，且$p(x),q(x)$的次数小于2。

令$x=\alpha$，得$p(\alpha)(a_0+a_1\alpha)=1$，所以：

$(a_0+a_1\alpha)^{-1}=p(\alpha)$。

对于一般情形，原则上需要应用辗转相除法来求取多项式$p(x)$，从而求取$GF(5^2)$中任意非0元$a_0+a_1\alpha$的逆元$p(\alpha)$，或用公式：$x^{-1}=x^{q-2}$来求取逆元，$(a_0,a_1)^{-1}=(a_0,a_1)^{5^2-2}=(a_0,a_1)^{23}$。但是，对这个具体的例子而言，可简单地应用待定系数法写出逆元公式。

设$p(x)=b_0+b_1x$，令：

$p(x)(a_0+a_1x)=(b_0+b_1x)(a_0+a_1x)=a_0b_0+2a_1b_1+(a_0b_1+a_1b_0)x=1$，得：

$a_0b_0+2a_1b_1=1$，$a_0b_1+a_1b_0=0$，

解得：$b_0=\dfrac{a_0}{a_0^2-2a_1^2}$，$b_1=-\dfrac{a_1}{a_0^2-2a_1^2}$。于是，得到逆元公式：

（4）$(a_0,a_1)^{-1}=(\dfrac{a_0}{a_0^2-2a_1^2},-\dfrac{a_1}{a_0^2-2a_1^2})$，$a_0^2-2a_1^2\neq 0$。

注意到，$(a_0,a_1)\neq 0$可得$(a_0,a_1)^2=(a_0^2+2a_1^2,2a_0a_1)\neq 0$，进而$a_0^2-2a_1^2\neq 0$。事实上，$(a_0,a_1)\neq 0$等价于$a_0^2-2a_1^2\neq 0$。

从这个例子可见，乘法、平方、逆元运算是很困难的。技术上，是通过选取适当的基来

简化运算，提高运行速度的。这是密码学应用中加速加密算法的一个重要方法。

密码学技术中最常用的有限域是 $GF(2^n)$，也称为二元域。可以证明，存在 $GF^*(2^n)$ 的一个生成元 θ，使 $\left\{\theta,\theta^{2^1},\theta^{2^2},\cdots,\theta^{2^{n-2}},\theta^{2^{n-1}}\right\}$ 成为 $GF(2^n)$ 在 Z_2 上的一个基，称为正规基。应用这个基，有很多运算将得到简化：

（1）乘法单位元为：$(1,1,\cdots,1)$；

（2）$(a_0,a_1,\cdots,a_{n-1})^2=(a_{n-1},a_0,a_1,\cdots,a_{n-2})$；

（3）$A^{-1}=A^{2^n-2}=A^{2^1+2^2+\cdots+2^{n-1}}=\left(A^{2^1}\right)\left(A^{2^2}\right)\cdots\left(A^{2^{n-1}}\right)$，需要 $n-2$ 次域乘法。

但是，乘法表达式却不这么简单了。

设 $A=(a_0,a_1,\cdots,a_{n-1}),B=(b_0,b_1,\cdots,b_{n-1})$，令：

$$C=AB=\left(\sum_{i=0}^{n-1}a_i\theta^{2^i}\right)\left(\sum_{j=0}^{n-1}b_j\theta^{2^j}\right)=\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}a_ib_j\theta^{2^i+2^j}=\sum_{k=0}^{n-1}d_k\theta^{2^k},$$

虽然，我们不能统一地写出 d_k 的表达式（需要根据 θ 的极小多项式化简，即：以 θ 为根的次数最小的多项式），但是，d_k 有一个特别的性质是很方便的。

$$C^2=(d_0,d_1,\cdots,d_{n-1})^2=(d_{n-1},d_0,d_1,\cdots,d_{n-2})$$

$$\begin{aligned}(AB)^2&=A^2B^2\\&=(a_0,a_1,\cdots,a_{n-1})^2(b_0,b_1,\cdots,b_{n-1})^2\\&=(a_{n-1},a_0,\cdots,a_{n-2})(b_{n-1},b_0,\cdots,b_{n-2})\end{aligned}$$

所以有：$(d_{n-1},d_0,d_1,\cdots,d_{n-2})=(a_{n-1},a_0,\cdots,a_{n-2})(b_{n-1},b_0,\cdots,b_{n-2})$。

可见，所有的 d_k 都具有同样的形式，于是有：

（4）$(a_0,a_1,\cdots,a_{n-1})(b_0,b_1,\cdots,b_{n-1})=(d_0,d_1,\cdots,d_{n-1})$，

如果：$d_0=g(a_0,a_1,\cdots,a_{n-1};b_0,b_1,\cdots,b_{n-1})$；

则：$d_1=g(a_1,\cdots,a_{n-1},a_0;b_1,\cdots,b_{n-1},b_0)$；

……………；

$d_k=g(a_k,\cdots,a_{n-1},a_0,\cdots,a_{k-1};b_k,\cdots,b_{n-1},b_0,\cdots,b_{k-1})$；

……………；

$d_{n-1}=g(a_{n-1},a_0,\cdots,a_{n-2};b_{n-1},b_0,\cdots,b_{n-2})$。

可见，平方运算只是移位，逆运算成为移位运算及乘法运算的复合，乘积各分量具有相同的函数表达形式，从而存储空间，运算速度，特别是硬件实现和速度将大大提高。

【例 2-10】 $p(x)=x^3+x^2+1$ 为 Z_2 上的不可约多项式，设 θ 为 $p(x)=0$ 在 $GF(2^3)$ 上的一个根，验证 $\theta,\theta^2,\theta^{2^2}$ 是 $GF(2^3)$ 在 Z_2 上的正规基，并求取 $GF(2^3)$ 在该基下的计算公式。

解 $1,\theta,\theta^2$ 是 $GF(2^3)$ 在 Z_2 上的一个基，

$\theta^4=\theta(\theta^3)=\theta(\theta^2+1)=\theta^3+\theta=\theta^2+\theta+1$。

设 $k_1\theta+k_2\theta^2+k_3\theta^4=0$，则 $k_3+(k_1+k_3)\theta+(k_2+k_3)\theta^2=0$，得到：

$\begin{cases}k_3=0\\k_1+k_3=0\\k_2+k_3=0\end{cases}$，于是：$k_1=k_2=k_3=0$，即 $\theta,\theta^2,\theta^{2^2}$ 线性无关，从而它是一个基。

$\theta^8=\theta$,

$\theta^3=\theta^4+\theta$,

$\theta^5=\theta^4+\theta^2$,

$\theta^6=\left(\theta^4+\theta\right)^2=\theta^8+\theta^2=\theta^2+\theta^4+\theta^2$,

$$
\begin{aligned}
AB&=\left(a_0\theta+a_1\theta^2+a_2\theta^4\right)\left(b_0\theta+b_1\theta^2+b_2\theta^4\right)\\
&=a_0b_0\theta^2+a_0b_1\theta^3+a_0b_2\theta^5+a_1b_0\theta^3+a_1b_1\theta^4+a_1b_2\theta^6+a_2b_0\theta^5+a_2b_1\theta^6+a_2b_2\theta^8\\
&=a_0b_0\theta^2+(a_0b_1+a_1b_0)\theta^3+a_1b_1\theta^4+(a_0b_2+a_2b_0)\theta^5+(a_2b_1+a_1b_2)\theta^6+a_2b_2\theta^8\\
&=(a_0b_1+a_1b_0+a_2b_1+a_1b_2+a_2b_2)\theta+(a_0b_0+a_0b_2+a_2b_0+a_2b_1+a_1b_2)\theta^2+\\
&\quad(a_0b_1+a_1b_0+a_1b_1+a_0b_2+a_2b_0)\theta^4\\
&=(d_0,d_1,d_2)
\end{aligned}
$$

令 $g\left(x_0,x_1,x_2;\ y_0,y_1,y_2\right)=x_0y_1+x_1y_0+x_2y_1+x_1y_2+x_2y_2$，则：

$d_0=g\left(x_0,x_1,x_2;\ y_0,y_1,y_2\right)$，

$d_1=g\left(x_1,x_2,x_0;\ y_1,y_2,y_0\right)$，

$d_2=g\left(x_2,x_0,x_1;\ y_2,y_0,y_1\right)$。

$$
\begin{aligned}
A^{-1}&=A^{2^3-2}=A^{2^1+2^2}=A^2A^{2^2}=A^2(A^2)^2\\
&=(a_0,a_1,a_2)^2\left((a_0,a_1,a_2)^2\right)^2\\
&=(a_2,a_0,a_1)(a_2,a_0,a_1)^2\\
&=(a_2,a_0,a_1)(a_1,a_2,a_0)\\
&=(\tilde{d}_0,\tilde{d}_1,\tilde{d}_2)\text{。}
\end{aligned}
$$

则：$\tilde{d}_0=g\left(a_2,a_0,a_1;\ a_1,a_2,a_0\right)=a_2a_2+a_0a_1+a_0a_0+a_1a_2+a_1a_0=a_0+a_2+a_1a_2$，

$\tilde{d}_1=g\left(a_0,a_1,a_2;\ a_2,a_0,a_1\right)=a_0+a_1+a_2a_0$，

$\tilde{d}_2=g\left(a_1,a_2,a_0;\ a_0,a_1,a_2\right)=a_1+a_2+a_0a_1$。

事实上，

$GF^*\left(2^3\right)=\left\{a_0+a_1\theta+a_2\theta^2\neq 0\mid a_0,a_1,a_2=0,1\right\}=\left\{1,\theta,1+\theta,\theta^2,1+\theta^2,\theta+\theta^2,1+\theta+\theta^2\right\}$，

$\theta^3+\theta^2+1=0$。

乘法、逆元运算表为：

表 2-1　乘法、逆元运算表

×	1	θ	$1+\theta$	θ^2	$1+\theta^2$	$\theta+\theta^2$	$1+\theta+\theta^2$
1	1	θ	$1+\theta$	θ^2	$1+\theta^2$	$\theta+\theta^2$	$1+\theta+\theta^2$
θ		θ^2	$\theta+\theta^2$	$1+\theta^2$	$1+\theta+\theta^2$	1	$1+\theta$
$1+\theta$			$1+\theta^2$	1	θ	$1+\theta+\theta^2$	θ^2
θ^2				$1+\theta+\theta^2$	$1+\theta$	θ	$\theta+\theta^2$
$1+\theta^2$					$\theta+\theta^2$	θ^2	1
$\theta+\theta^2$						$1+\theta$	$1+\theta^2$
$1+\theta+\theta^2$							θ
逆元	1	$\theta+\theta^2$	θ^2	$1+\theta$	$1+\theta+\theta^2$	θ	$1+\theta^2$

2.4 习题

1. 证明例 2-3 的结论。
2. 证明例 2-4 的结论。
3. 证明定理 2-3 的推论 2。
4. 求阶为 4，7，8 的群。
5. 求 Z_5 上所有不可约多项式，并找出这些多项式中在 Z_3 上可约的多项式。
6. 验证下列多项式是 Z_2 上的不可约多项式：

（1） x^4+x+1；

（2） $x^5+x^4+x^2+x+1$；

（3） $x^6+x^4+x^2+x+1$；

（4） $x^7+x^6+x^5+x^3+x^2+x+1$；

（5） $x^8+x^4+x^3+x^2+1$；

（6） $x^9+x^4+x^3+x+1$；

（7） $x^{10}+x^7+x^4+x^3+1$。

7. 证明：F 的本原元的个数为 $\varphi(q-1)$，其中 $q=\#(F)$。
8. 构造有限域 $GF(2^2)$，$GF(2^3)$，$GF(3^2)$，$GF(2^4)$，并求其所有本原元。

2.5 实验题

1. 编写求取 Z_p 上不可约多项式的计算机程序。
2. 编写有限域 $GF(2^n)$ 运算的计算机程序。

第3章　密码学基本概念

3.1　密码技术发展简介

密码是一个古老的话题。早在四千多年前，古埃及人就开始使用密码来保密传递的消息。作为保密信息的手段，密码技术本身也处于秘密状态，基本上局限于军事目的，只为少数人掌握和控制，所以，它的发展受到了限制。

在第一次世界大战之前，密码技术的进展很少见诸于世，直到 1918 年，William F.Friedman 的论文“重合指数及其在密码学中的应用”（The Index of Coincidence and Its Applications in Cryptography）发表时，情况才有所好转。

1949 年，香农（Claude Shannon）的奠基性论文“保密系统的通信理论”（The Communication Theory of Secrecy Systems）在《贝尔系统技术杂志》上发表，奠定了密码学理论基础。

此后，直到 1967 年，密码学文献几乎空白。

1967 年，戴维・卡恩（David Kahn），收集整理了第一次世界大战和第二次世界大战的大量史料，创作出版了《破译者》（The Codebreakers），为密码技术的公开化、大众化拉开了序幕。此后，密码学的文献大量涌现。

20 世纪 70 年代，是密码学发展的重要时期，有两件重大事件发生。

其一，美国国家标准局（NBS，即现在的国家标准与技术研究所 NIST）开始数据加密标准（Data Encryption Standard）的征集工作。1975 年 3 月 17 日，NBS 在 Federal Register 上公布了一个候选算法，1976 年 11 月 23 日，该算法被正式确认为联邦标准 DES，并授权在政府通信中使用，此后 DES 被多个部门和标准化机构采纳为标准，甚至成为事实上的国际标准，1998 年正式退役。

其二，1976 年 11 月，Diffie 与 Hellman 的革命性论文“密码学新方向”（New directions in crypotography）发表，开辟了公开密钥密码学的新领域，成为现代密码学的一个里程碑。1978 年，R.L.Rivest，A.Shamir 和 L.Adleman 实现了 RSA 公钥密码体制，它成为公钥密码的杰出代表和事实标准。

1969 年，哥伦比亚大学的 Stephen Wiesner 首次提出“共轭编码”（Conjugate coding）的概念。1984 年，Bennett. Charles H., Brassard. Gille 在 Wiesner 的思想启发下，首次提出了基于量子理论的（现称为）BB84 协议，从此量子密码理论宣告诞生。

量子密码不同于以前的密码技术，是一种可以发现窃听行为、安全性基于量子定律的密码技术，可以抗击具有无限计算能力的攻击，有人甚至认为，在量子计算机诞生之后，量子密码技术可能成为惟一的真正安全的密码技术。

1985 年，N.Koblitz 和 V.Miller 把椭圆曲线理论应用到公钥密码技术中，在公钥密码技术中取得重大进展，成为公钥密码技术研究的新方向。

密码技术的另一个重要方向——流密码（也称序列密码）理论也取得了重要的进展，1989年 R.Mathews，D.Wheeler，L.M.Pecora 和 Carroll 等人把混沌理论引入到流密码及保密通信理论中，为序列密码理论开辟了一条新的途径。

1997 年 9 月 12 日，美国国家标准与技术研究所 NIST 开始征集新一代数据加密标准来接任即将退役的 DES，2000 年 10 月 2 日，由比利时密码学家 Joan Daemen，Vincent Rijmen 提交的 Rijndael 算法被确定为 AES 算法。

2000 年 1 月，欧盟启动了新欧洲数据加密、数字签名、数据完整性计划 NESSIE，旨在提出一套强壮的包括分组密码，流密码，散列函数，消息认证码（MAC），数字签名和公钥加密的密码标准，使欧洲工业界保持密码学研究领域的领先地位，2002 年底最后确定各类标准算法。

计算机、电子通信技术的飞速发展使现代密码学及其应用得到了重大发展。

计算机网络的广泛应用，产生了大量的电子数据，这些电子数据需要传输到网络的各个地方并存储。这些数据有的具有重大的经济价值，有的关系国家、军队或企业的命运甚至生死存亡。对于这些数据，有意的计算机犯罪和无意的数据破坏都会造成不可估量的损失。如：国家机密、企业经济信息、银行电子资金业务等等，它们的任何一点泄露和差错都将会使国家安全，企业经营受到巨大的伤害。

如果把这些威胁称为攻击，那么，简单地讲，攻击可分为两种形式：一种称为被动攻击，就是非法地从传输信道上截取信息，或从存储载体上拷贝信息。另一种称为主动攻击，就是伪造或对传输或存储的数据进行破坏、删除、篡改等。对于这些犯罪行为，虽然相应的法律已逐步建立,但是，由于这种犯罪形式的特殊性，对于它的监督和量刑都是很困难的。因而，除了不断完善相应法律和监督措施，我们更需要自我保护。理论和事实都说明，密码技术是一种经济、实用而有效的方法。这也是密码技术得到快速发展和广泛应用的原因。

经典的密码学是关于加密和解密的理论，主要用于通信保密。在今天，密码学已得到了更加深入、广泛的发展。其内容已不再是单一的加解密技术，已被有效、系统地用于电子数据的保密性、完整性、真实性和不可否认性等各个方面。

保密性就是对数据进行加密，使非法用户无法读懂数据信息，而合法用户可以应用密钥读取信息。

完整性是对数据完整性的鉴别，以确定数据是否被非法篡改，保证合法用户得到正确、完整的信息。

数据的真实性是数据来源的真实性、数据本身的真实性鉴别，可以保证合法用户不被欺骗。

不可否认性是真实性的另一个方面，有时候，我们不但要确定数据来自何方，而且还要求数据源不可否认发送的数据。

现代密码技术的应用已深入到信息安全的各个环节和对象。主要技术有：数据加密，密码分析，数字签名，信息鉴别，零知识认证，秘密共享等等。密码学的数学工具也更加丰富，概率统计、数论、组合、代数、混沌、椭圆曲线等，现代数学的许多领域都有密码学的足迹。

另一个值得一提的就是近年来发展迅速的信息隐藏技术。这是一种将需要保密的信息隐藏在公开信息中来保密、传输的技术，所以可以称为秘密的信息嵌入技术，主要基于不在意

信息传输的思想。相对而言，密码技术是一种公开的信息嵌入、刻意传输技术。如果把密钥视为秘密隐藏信息的秘密，那么，广义地说，密码技术也是一种信息隐藏技术。这种信息隐藏的秘密由密钥控制。狭义隐藏信息的秘密由算法控制。如果算法暴露或公开，那么信息也就无法隐藏了，所以，信息隐藏技术的另一个重要环节是不在意传输。将信息隐藏在平凡，经常性的公开信息（如广播、电视、图片等）中进行传输，使窃密者无从下手，以此达到逃避信息窥测的目的。数字水印，数字指纹等技术是信息隐藏技术的典型方法，已被广泛地应用于数字产品的产权保护上。

3.2 密码系统的概念

密码技术的基本功能是保密通信的信息。经典的保密通信模型如图 3-1：

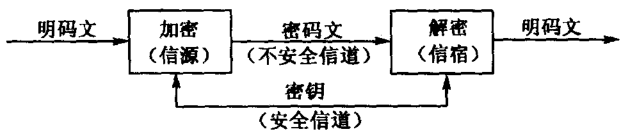


图 3-1　经典保密通信模型

我们将消息的发送者称为信源，消息的接收者称为信宿，原始的消息称为明码文（简称明文），经过变换（称为加密）的消息称为密码文（简称密文）。用来传输消息的通道称为信道。通信过程中，信源为了和信宿通信，首先要选择适当的加密参数 k，称为密钥，并把它通过安全的信道（如：武装信使等）送给信宿，通信时，把明文 m 通过加密方法（变换）E_k 加密成密文 $c=E_k(m)$，通过普通（不安全）信道发送给信宿，信宿应用信源从安全信道送来的密钥 k，通过解密变换 D_k 解密密文 c，恢复出明文 $m=D_k(c)=D_k(E_k(m))$。

通过这个模型，可以给出密码体制的基本概念，一个密码体制可以有如下几个部分：

（1）所有可能的明文的集合 P，称为明文空间；

（2）所有可能的密文的集合 C，称为密文空间；

（3）所有可能的密钥的集合 K，称为密钥空间；

（4）加密算法：E: $P\times K\rightarrow C$，$(m,k)\mapsto E_k(m)$，

解密算法：D: $C\times K\rightarrow P$，$(c,k)\mapsto D_k(c)$，

对 $\forall m\in P, k\in K$，有 $D_k(E_k(m))=m$。

五元组 (P,C,K,E,D) 称为一个密码体制。

一个完整的保密通信系统是由一个密码体制，一个信源，一个信宿，还有一个攻击者或破译者构成。且假设：第一，对所有密钥，加解密算法迅速有效，常常需要实时使用；第二，体制的安全性不依赖于算法的保密，只依赖于密钥的保密。这是荷兰密码学家 Kerchhoff 提出的密码学基本假设，于是称为 Kerchhoff 原则。也就是假定，除密钥以外，破译者掌握密码算法的所有内容和细节，系统的安全性完全依赖于密钥的保密。

完整的保密通信系统如图 3-2 所示。

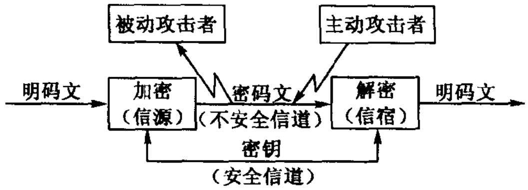


图 3-2 完整的保密通信系统

经典的密码体制中，值得注意的是加密密钥与解密密钥是相同的，或者可以简单相互推导。也就是说：知道了加密密钥，也就知道了解密密钥；知道了解密密钥，也就知道了加密密钥。所以，加、解密密钥必须同时保密。这种密码体制称为对称密码体制，也称单钥密码体制。最典型的是美国数据加密数据加密标准 DES，应该说数据加密标准 DES 是单钥体制的最成功的例子。现代密码学修正了密钥的对称性，1976 年，Diffie，Hellman 提出了公开密钥密码体制（简称公钥体制），它的加密和解密密钥是不同的，也是不能（在有效的时间内）相互推导的。所以，可称为双钥密码体制。它的产生，是密码学革命性的发展，它一方面为数据的保密性、完整性、真实性提供了有效方便的技术。另一方面，科学地解决了密码技术的瓶颈——密钥的分配问题。

第一个有效的公钥体制是 1977 年由 Rivest，Shamir，Adleman 提出的，称为 RSA 公钥体制，其安全性基于整数的因子分解的困难性，RSA 公钥体制已得到了广泛的应用。其后，诸如基于背包问题的 Merkle-Hellman 背包公钥体制，基于有限域上离散对数问题的 ElGamal 公钥体制，基于椭圆曲线的密码体制等等公钥体制不断出现，密码学得到了蓬勃的发展，公钥体制的提出奠定了密码学的一个新的里程碑。

公钥体制的基本方法是：每个用户有两个密钥，一个用于加密，称为公开密钥，一个用于解密，称为秘密密钥。密码体制的发布由信源实施，他将加密密钥公开，使得任何需要和其通信的人都可以取得，从而可以向其发送加密信息。相反，他将解密密钥保密，用于解密收到的密文。攻击者虽然也可以得到加密密钥，但由于不能由此推导出解密密钥而无法解密密文。

可见，公钥密码体制将密钥的秘密传递变成了公开发布，成功地解决了密钥分配问题。公钥密码体制还为数字签名提供了有效的方法。

以上仅是一个简单的示意模型。由于保密性、真实性、密钥分配及防止各种可能攻击等具体需要，实际模型是复杂的。

3.3 密码分析

加密与破译始终是一对孪生的问题，破译也叫密码分析。原则上来说，密码分析是攻击者为了窃取机密信息所做的事情，但是，这也是密码体制的设计者的工作。当然，设计者的目的是为了分析体制的弱点，以期提高体制的安全强度。

一般地，如果仅根据密文就可以推算出明文或密钥，则称这个密码体制是可破译的。

一个密码体制是安全的，其总的前提就是假设密码分析者已经知道了密码体制的算法，体制的安全性仅依赖于密钥的保密，即 Kerchhoff 原则。

密码分析大体上可分为三类：穷举法，统计法，系统分析法。

所谓穷举法，就是对可能的密钥或明文的穷举。穷举密钥时，用可能密钥解密密文，直到得到有意义的明文，确定出正确的密钥和明文。穷举明文，就是将可能的明文加密，将所得密文与截取的密文对比，从而确定正确的明文。这一方法主要用于公钥体制和数字签名。阻止穷举的方法有：增加密钥的长度，在明文，密文中增加随机冗余信息等等。

所谓统计法，就是根据明文，密文，密钥的统计规律分析密文的统计规律达到破译密码体制的方法。例如，在经典的换位密码，置换密码体制中，可根据单字母，双字母，三字母等的频率分布，最后确定密钥。阻止统计法的方法是：把密文和明文的统计特性扩散到整个密文，使密文不呈现任何统计特性，而呈现出极大的随机性。

系统分析法，就是根据掌握的明、密文的有关信息，应用加、解密算法的有关性质，求解密钥、明文的方法。原则上，受到系统分析破译的密码体制已完全不能使用。

理论上，除一文一密的密码体制外，没有绝对安全的密码体制。所以，称一个密码体制是安全的，一般是指密码体制在计算上是安全的，即：密码分析者为了破译密码，穷尽其时间和存储资源仍不可得，或破译所耗费的成本已超出了因破译密码而获得的收益。

根据密码分析者对明、密文掌握的程度，攻击主要可分为三种：

（1）唯密文攻击：

指密码分析者仅根据截获的密文进行的密码攻击。

（2）已知明文攻击：

指密码分析者已经掌握了一些相应的明、密文对，根据这些明、密文对对密码体制进行的攻击。

（3）选择明文攻击：

密码分析者可以选择一些明文，并可取得相应的密文。例如：公开密钥密码体制中，分析者可以用公开的密钥加密他任意选定的明文。这种攻击称为选择明文攻击。

一个密码体制是安全的一般是指在以上三种攻击下的安全性。并且这三种攻击的强度是渐增的，对公钥体制的攻击是一种选择明文攻击，因而公钥体制抗选择明文攻击是体制安全的必要条件。

3.4 数字签名与认证

数字签名是电子信息的特殊产物，最早被建议用来对禁止核试验条律的验证。禁止核试验条律的缔约国为了监测对方的核试验，需要把地震测试仪放在对方的地下，而把测试的数据送回。自然这里有一个矛盾：东道主需要检查发送的数据是否仅为规定的测试数据；而检测方需要保证送回的数据确实是检测到的数据，东道主没有篡改。对这些数据，根本的要求是：东道主只能读，不能改。这就是数字签名的本质。

数字签名是保证电子数据真实性的有效手段，传统的手写签名一直是人们对纸质文件内容真实性、不可否认性采用的一种措施。对签名的目的和要求可归纳为如下几点：

（1）签名是不能伪造的。签名是签名者对文件内容合法性的认同、证明和标记，其他人的签名无效；

（2）签名是不可抵赖的。这是对签名者的约束，签名者不可否认自己的认同、证明。

（3）签名是不可改变的。文件签名后签名是不可改变的，这就保证了签名的真实性、可靠性。

（4）签名是不可以重复使用的。签名和文件内容是一个整体，签名保证了文件内容的真实性，因而，签名是不可复制、重复使用的，这就相当于说文件内容是不可以伪造的。

（5）签名是容易验证的。对于签名的文件，一旦发生纠纷，可以准确、有效地进行验证。

但是，事实上，传统的手写签名并没有有效地、真正地做到这些：

（1）签名与文件是物理地联系在一起的，如果采用物理方法进行剪切、拼接，就可以移花接木，伪造文件内容，当然，这需要高超的物理拼接技术。但是，对于具有多页的文件，这种替换可能是简单的。

（2）手写签名的鉴别是通过视觉对手写图形的辨认来确定的，因而其精确性无从保证，模仿伪造者也就有机可乘了。

数字签名却很好地解决了这些问题。数字签名可以用秘密密钥密码体制实现，也可以用公开密钥密码体制来实现。

事实上，签名的基本原理很简单。

如果应用公开密钥体制，则数字签名就是加、解密过程的颠倒使用。签名者应用加密密钥对文件加密——签名，验证者对文件解密——验证。签名者将签名（加密）密钥保密，验证（解密）密钥公开。这样，因为只有签名者才有加密密钥，所以，验证者对照明、密文即可确定签名文件的真实性。

用秘密密钥实现的数字签名一般需要一个安全的、值得信赖的仲裁者的参与。设仲裁者与每一个用户共享一个秘密密钥，则对称密钥体制的数字签名的一般模型如图 3-3 所示。

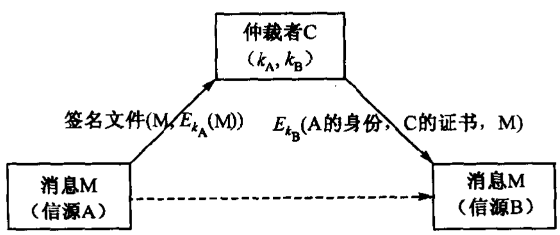


图 3-3　对称密钥体制的数字签名模型

信源 A 用密钥 k_A 加密（签名）消息 M，发送给仲裁者 C，C 用 k_A 解密（验证）消息确实来源于 A，然后为 A 作一个证书，连同 A 的身份信息，消息 M 一起用 k_B 加密，发送给 B。由于只有自己和 C 拥有 k_B，所以 B 确信消息来自 C，通过 C 的证书及 A 的身份信息，B 确知消息来自 A。

无论是公钥体制，还是秘钥体制实现的签名，都应用了两个不可互相推导的密钥，一个用作签名，一个用作验证。

签名前，为了提高效率，一般需要对文件进行一定的预处理——应用散列（Hash）函数提取文件摘要，最后对文件摘要进行签名。更一般地，有时候还需要对文件进行加密。

散列函数是数字签名的一个重要辅助工具，应用散列函数可以生成一个文件的具有固定长度的摘要，从而使数字签名可以有效、快速地签名任意长度的文件。

散列函数以文件作为输入，输出固定长度的值，一般地，对文件的任意微小的改动，都会改变文件的散列值，因而秘密的散列函数可以用来检测病毒对文件的破坏。例如，存储文件时，可以用一个秘密散列函数生成一个散列值置于文件的尾部，查毒时，应用同一散列函数生成文件（除去尾部固定长度的散列值）的散列值，与文件尾部进行比较，即可确定文件是否染毒。

对于以上描述的简单的公钥模型或秘钥模型，都可能发生一种欺骗，即：签名的重复使用。例如，对于一张数字签名的电子支票，如果没有适当的措施，就可以重复使用，重复使用的后果是显而易见的。阻止这种欺骗的方法是：在签名中包含日期和时间标记。

数字签名不是完美的，有各种各样的欺骗存在，虽然有一些欺骗不是通过签名本身可以解决的，但是，对于大部分欺骗，通过修正数字签名的算法和方案是可以对付的。所谓道高一尺，魔高一丈，随着研究的不断深入，数字签名的理论、应用将会不断完善、成熟。

3.5 计算复杂性理论

计算复杂性理论是密码安全性理论的基础。它研究分析问题的有效解法，提供求解问题所需的运算次数。从而给出问题求解困难性的数量指标。

加密算法，解密算法所需的计算量是密码算法运行效率的一个重要指标，而密码分析所需的计算量则是密码体制安全性的生命指标。如果用 n 表示问题的大小，或输入的长度，则计算复杂性可用两个参数来表示，即：运算所需的时间 T 和存储空间 S，它们都是 n 的函数，于是可记为：$T(n)$，$S(n)$。

如果 $T(n)=O(n^c)$ $(c>0)$，则称该算法的时间是多项式阶的，如果 $T(n)=O(a^{P(n)})$ $(a>0)$，则称算法时间是指数阶的，其中 $P(n)$ 是 n 的一个多项式（非常数）。对于指数阶时间的算法，适当增大 n，计算将变成不可能的。一般认为，如果破译一个密码体制所需的时间是指数阶的，则它是计算上安全的。

空间复杂性与时间复杂性往往是可以相互转换的。

例如，可以预算某些明、密文对，分析时只须查询即可，这样计算的时间就转换成了存储的空间。

算法可分为确定性算法和非确定性算法，确定性算法的每一步的操作结果都是确定的，其计算时间就是完成这些确定步骤所需的时间。不确定性算法的某些操作结果是不确定的，使算法成功的操作序列中，所需时间最少的序列所需时间就是该不确定性算法的计算时间。

用确定性算法可以在多项式时间内求解的问题称为 P 问题。在多项式时间内可以用非确定性算法求解的问题称为 NP 问题。显然 NP 问题包含 P 问题，但是，P 问题是否包含 NP 问题，不得而知。P≠NP 目前仍然是一个开问题。

如果问题 G 可以在多项式时间内用确定性算法转化为问题 L，而 L 的解可在多项式时间内用确定性算法转化为 G 的解，则称问题 G 可归约为问题 L。

如果某类 NP 问题中的任何一个问题都可归约为问题 L，且 L 本身也是 NP 问题，则称 L 是一个 NP 完全问题。

对于一个 NP 问题，不存在任何已知的确定性算法在多项式时间内求解该问题，所以，如果能够找到一个计算序列，作为解密算法，那么，密码分析者在不知道计算序列的情形下

求解问题——称为客观求解——成为计算上不可能的了。可见，NP 问题适合用来构造密码体制。Diffie 和 Hellman 指出，NP 完全问题更适合构造密码体制。

现已证明，正整数的因数分解问题是一个 NP 问题。对于一个正整数 N，已知的最快的客观算法的算术次数是：$e^{\sqrt{\ln N \ln\ln N}}$。同样，背包问题，Hamilton 回路问题等都是 NP 问题。目前已知的 NP 问题有 300 多个。

3.6 传统密码举例

密码方法经过了手工——机械——电子几个阶段。现代电子阶段密码的基本原理是基于复杂的数学运算或数学难题的；而手工阶段、机械阶段密码的基本原理大体上可以分成两类，一类是基于符号替换的替换密码，另一类是基于书写位置交换的换位密码。

1．替换密码

任何文字都有一个符号表，传统密码方法中，有许多是以符号表的置换来达到掩盖明文信息的。典型的如：移位密码、单表置换密码、多表置换密码等等。下面以英文为例简要说明。

（1）移位密码：将英文字母循环向前或向后移动一个固定位置。

例如向后移动 3 个位置，即对字母表作置换（不分大小写）：

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

设明文为：public keys，则经过以上置换就变成了：sxeolf nhbv。

如果将 26 个英文字母进行编码：A→0，B→1，…，Z→25，则以上加密过程可简单地写成：

明文：$m=m_1m_2\ldots m_i\ldots$，则有

密文：$c=c_1c_2\ldots c_i\ldots$，其中：$c_i=$（m_i+3 mod 26），i=1,2,…。

这一方法，有史记载，最早约在公元前 50 年，被罗马大帝朱利叶斯·凯撒（Julius Caesar）使用，所以也称为凯撒密码。

一般的移位密码算法为：

明文：$m=m_1m_2\ldots m_i\ldots$；

密钥：k；

密文：$c=c_1c_2\ldots c_i\ldots$，$c_i=$（m_i+k mod 26），i=1,2,…。

（2）单表置换密码

单表置换密码就是根据字母表的置换对明文进行变换的方法，例如，给定置换表为：

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
H	K	W	T	X	Y	S	G	B	P	Q	E	J	A	Z	M	L	N	O	F	C	I	D	V	U	R

明文：public keys，则有

密文：mckebw qxuo。

关于置换表的构造可以有各种途径，主要是为方便记忆。

如果字符的替换已经超出了字符集，则密码也称为代换密码，例如，将英文字符替换成希腊字符：

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
α	β	χ	δ	ε	ϕ	γ	η	ι	φ	κ	λ	μ	ν	ο	π	θ	ρ	σ	τ	υ	ϖ	ω	ξ	ψ	ζ

则明文：public keys，变换为密文：πυβλιχ κεψσ。

单表置换密码的本质是改变字符表，将英文字符对应地转换成希腊字母就是典型，所以本质上没有加密。明文的很多特性都没有被掩盖，例如，自然语言的字母使用频率，双字母、三字母组合规律等都没有改变，所以，可以简单地应用频率分析及自然语言的规则破译。为了弥补这种缺陷，人们又提出了多表置换的密码方法，这很好地解决了频率问题。

（3）多表置换密码

用多表置换密码加密时，同一个字符具有不同的密文，从而改变了单表置换中密文的惟一性。

顾名思义，多表置换中使用了多张字符置换表。例如，置换使用了 4 张置换表，则明文必须分成 4 个字符的分组，对每一组的 4 个字符，应用这 4 张表进行置换。

例如：4 张表为：

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
A	T	U	B	Q	W	L	F	D	O	H	K	S	V	C	E	I	R	J	X	G	Y	Z	M	N	P
H	K	W	T	X	Y	S	G	B	P	Q	E	J	A	Z	M	L	N	O	F	C	I	D	V	U	R

则明文：publ　icke　ys，变换为密文：satelihxby。

维吉利亚密码（Vigenere）是多表置换密码的著名代表，它的置换表都是移位置换表，移位的次数由密钥控制。例如：

设明文：public keys，密钥为：crypt，

则移位次数分别为 2，17，24，15，19，

置换表为：

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
c	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
r	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
p	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
t	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S

明文分成 5 个字符的分组：publi　ckeys

密文为：rlzab ebcnl。

请用数学表达式表示维吉利亚密码加、解密算法。

更一般的替换是多字母替换。它可以增大替换表的大小，从而增加密钥量。

维吉利亚密码虽然改变了字符的频率，但它本质上是分组移位密码，各分组相同位置的

字符仍然具有相同的密文，所以，对密文移位对照，计算重合指数可以确定其分组大小，然后按照相同位置，将分组组成具有相同移位次数的新的分组，进一步对各分组移位差异进行配对试验，消除移位差最后得到同一个移位置换的移位置换密文，进而轻易破译（请参看有关参考书）。

但是，对于一般的多表置换密码，如果置换表足够多（最多 26！张），即使现代密码分析技术也不能够破译。但维吉利亚密码最多只有 26 张，因而重合指数方法总是能够破译的。

2．换位密码

换位密码本质上也是一种置换密码，但它置换的不是字符，而是书写的位置。

如：将明文逐行写在一张格纸上，再按列的方式写出密文，则得到换位密文。

设明文为：cryptography is an applied science，写入 5 列的格纸，

c	r	y	p	t
o	g	r	a	p
h	y	i	s	a
n	a	p	p	l
i	e	d	s	c
i	e	n	c	e

则密文为：cohniirgyaeeyripdnpaspsctpalce。

将明文写在一个纸带上，卷在一根圆柱上，再按纵向读出文字作为密文也是一种换位密码，圆柱的半径就是密钥。

我国古代文学中的藏头诗、回文诗等采用的就是换位密码方法。

换位密码的数学表达式可写成：

设明文为：$m=m_1m_2\ldots m_i\ldots$，

则密文为：$c=c_1c_2\ldots c_i\ldots$，$c_i=m_{\pi^{-1}(i)}$，$i=1,2,\ldots$，其中置换表为：

$$\begin{matrix} 1 & 2 & 3 & 4 & \ldots & i & \ldots & n \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ \pi(1) & \pi(2) & \pi(3) & \pi(4) & \ldots & \pi(i) & \ldots & \pi(n) \end{matrix}$$

是一个$\{1,2,\ldots,i,\ldots,n\}$到$\{1,2,\ldots,i,\ldots,n\}$的一个置换。

例如，对于以上 5 列的格纸，有：

$$\begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 & 25 & 26 & 27 & 28 & 29 & 30 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 1 & 6 & 11 & 16 & 21 & 26 & 2 & 7 & 12 & 17 & 22 & 27 & 3 & 8 & 13 & 18 & 23 & 28 & 4 & 9 & 14 & 19 & 24 & 29 & 5 & 10 & 15 & 20 & 25 & 30 \end{matrix}$$

于是有：$\pi(i)=[(i-1)/5]\times5+1+(i-1) \bmod 5$，$i=1,2,\ldots,n$。

3.7 习题

1．用自己的观点划分密码理论和技术发展的阶段。

2．密码技术有哪些主要用途？

3．用信息论的观点阐述 Kerchhoff 原则的思想。

4．分析有仲裁者的数字签名中存在的欺骗和攻击。

5. 对大数因子分解的筛法，幂-模算法，模逆算法，试计算其时间、空间复杂性。

6. 用置换(abcdefghijklmnopqrstuvwxyz)→(iamstudenhcroxygwpbfjklqvz)加密消息：

when i started to work on this topic some people had difficulties seeing the relationship between geography and the internet although it is rather simple one can already find parallels between geography and the internet considering the language that is used to describe the internet if you came across this site using netscape navigator or microsoft internet explorer you are using an elementary tool whose name comes from the terminology of ancient expeditions

7. 破译单表置换密文：

SIC GCBSPNA XPMHACQ JB GPYXSMEPNXIY JR SINS MF SPNBRQJSSJBE JBFMPQNSJMB FPMQ N XMJBS N SM N XMJBS H HY QCNBR MF N XMRRJHAY JBRCGZPC GINBBCA JB RZGI N VNY SINS SIC MPJEJBNA QCRRNEC GNB MBAY HC PCGMTCPCD HY SIC PJEISFZA PCGJXJCBSR SIC XNPSJGJXNBSR JB SIC SPNBRNGSJMB NPC NAJGC SIC MPJEJBNSMP MF SIC QCRRNEC HMH SIC PCGCJTCP NBD MRGNP N XMRRJHAC MXXMBCBS VIM VJRICR SM ENJB ZBNZSIMPJOCD GMBSPMA MF SIC QCRRNEC

8. 破译单表置换密文：

OLUMLUXCXGLUDNLQLIUCNICMLLKRXUDLPDXGTUDFIYYQLGLUQXRLUIDTKZCQCDL
SDNLULIULIDYLICDDHXXRLKCXFUVLGLUCTXKCHTDNKFSLUXFCVXSSLUVTIYGLUCTXK
CMICLPXKDNLCLIKPXDNLURUXRUTLDIUQTSRYLSLKDIDTXKCOLUMLUXCLGXYFDTXKNI
CVXKDTKFLPXGLUDNLQLIUCIKPTKDLUXRLUIMTYTDQNICMLLKRUXMYLSIDTVIKFSMLU
XEPUIEDRUXRXCIYCNIGLMLLKTCCFLPVXKVLUKTKZICRLVDCXEKLHXULJDLKPLPEFKV
DTXKIYTDQ

第4章　对称密码算法

4.1　美国数据加密标准 DES

4.1.1　历史

1972 年，美国国家标准局（NBS）制定了一个保护计算机和通信开发计划，准备开发一个标准的密码算法，来规范密码技术应用的混乱局面。要求这个算法能够被测试和验证，不同设备间可以互相操作，且实现成本低。

1973 年 5 月 15 日，NBS 在《联邦记事》（Federal Register）上公布了征集保护传输、存储系统中计算机数据密码算法的请求及有关技术、经济和兼容性要求：

（1）算法具有较高的安全性，安全性仅依赖于密钥，不依赖于算法；

（2）算法完全确定，易于理解；

（3）算法对任何用户没有区别，适用于各种用途；

（4）实现算法的电子器件必须很经济；

（5）算法必须能够验证、出口。

但是，由于此前公众对密码知识的匮乏，所以，虽响应激烈，但提交的方案均不理想。

1974 年 8 月 27 日，NBS 再次发表征集公告，IBM 公司提交了一个良好的候选算法。这一个算法是 LUCIFFER 密码算法的改进，LUCIFFER 密码算法是 Horst Feistel 在 1971 年设计的。

1975 年 3 月 17 日，NBS 在《联邦记事》上公布了这一算法的细节。随后发表通知，征求公众的评论。

尽管受到了强烈的批评和责难，1976 年 11 月 23 日，这一算法还是被采纳为联邦标准，并授权在非密级的政府通信中使用。该标准的正式文本 FIPS PUB46（DES）在 1977 年 1 月 15 日公布，其他文件 FIPS PUB81（DES 工作方式），FIPS PUB74（DES 实现和使用指南），FIPS PUB112（DES 口令加密），FIPS PUB113（DES 计算机数据鉴别）等联邦信息处理标准也相继于 80 年代初公布。

1981 年，美国国家标准研究所（ANSI）批准 DES 作为私营部门的数据加密标准，称为 DEA，有关文件：ANSIX3.92（DEA），ANSIX3.106（DEA 工作方式），ANSIX3.105（网络加密标准）等相继公布。

国际标准化组织（ISO）将 DES 改名为 DEA-1 准备作为国际标准。但是，1997 年，ISO 的信息处理 97 技术委员会（TC97）决定只接受密码算法公开注册，而不再进行密码算法的标准化工作，这样 DES 最终没能成为国际标准。

4.1.2　算法

DES 是一种明文分组为 64 比特，有效密钥 56 比特，输出密文 64 比特的，具有 16 轮

迭代的分组对称密码算法，DES 由初始 *IP* 置换，16 轮迭代，初始 *IP* 逆置换组成。

1．DES 的基本运算

（1）初始置换 *IP*

IP 是一个 64 明文比特的一个排列，若记明文为 $x = x_1x_2\cdots x_{64}$，其中 $x_i = 0,1$ ，$i = 1,2,\cdots,64$，则 $IP(x) = x_{58}x_{50}\cdots x_{15}x_7$，即对明文位置作如图 4-1 所示的置换：

1	2	3	4	5	6	7	8		58	50	42	34	26	18	10	2
9	10	11	12	13	14	15	16		60	52	44	36	28	20	12	4
17	18	19	20	21	22	23	24	*IP*	62	54	46	38	30	22	14	6
25	26	27	28	29	30	31	32	⇒	64	56	48	40	32	24	16	8
33	34	35	36	37	38	39	40		57	49	41	33	25	17	9	1
41	42	43	44	45	46	47	48		59	51	43	35	27	19	11	3
49	50	51	52	53	54	55	56		61	53	45	37	29	21	13	5
57	58	59	60	61	62	63	64		63	55	47	39	31	23	15	7

图 4-1　*IP* 置换

可见，置换的构造如图 4-2 所示：

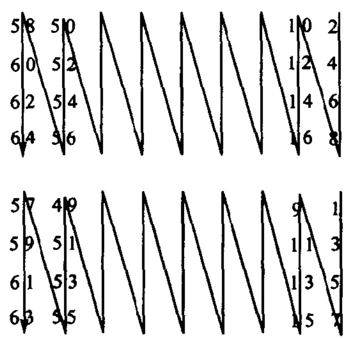


图 4-2　*IP* 置换的构造规则

逆序先取偶数位，4 个一组按列排列，再逆序取奇数位，4 个一组排列。

用数学表达式可写成：

$$IP(x) = x_{58}x_{60}\cdots x_{\pi(i)}\cdots x_{15}x_7，\ \pi(i+1) = 58 - \left[\frac{i}{32}\right] + 2\left[\frac{i \mod 32}{8}\right] - 8(i \mod 8)，$$

$i = 0,2,\cdots,63$，其中 [ ] 表示整数部分 。

IP 的逆置换如图 4-3 所示，$IP^{-1}(x) = x_{40}x_8\cdots x_{\pi^{-1}(i)}\cdots x_{57}x_{25}$，即：

$$\pi^{-1}(i+1) = 8 + 32\ \left((i+1) \mod 2\right) - \left[\frac{i}{8}\right] + 4\ \left((i - i \mod 2) \mod 8\right), i = 0,2,\cdots,63。$$

（2）E-运算

E-运算是扩位运算，将 32 比特扩展为 48 比特，用方阵形式可以容易地看出其扩位方法，见图 4-4：

1	2	3	4	5	6	7	8		40	8	48	16	56	24	64	32
9	10	11	12	13	14	15	16		39	7	47	15	55	23	63	31
17	18	19	20	21	22	23	24	IP^{-1}	38	6	46	14	54	22	62	30
25	26	27	28	29	30	31	32	$\Rightarrow$	37	5	45	13	53	21	61	29
33	34	35	36	37	38	39	40		36	4	44	12	52	20	60	28
41	42	43	44	45	46	47	48		35	3	43	11	51	19	59	27
49	50	51	52	53	54	55	56		34	2	42	10	50	18	58	26
57	58	59	60	61	62	63	64		33	1	41	9	49	17	57	25

图 4-3 IP^{-1} 置换

1	2	3	4		32	1	2	3	4	5
5	6	7	8		4	5	6	7	8	9
9	10	11	12		8	9	10	11	12	13
13	14	15	16	E $\Rightarrow$	12	13	14	15	16	17
17	18	19	20		16	17	18	19	20	21
21	22	23	24		20	21	22	23	24	25
25	26	27	28		24	25	26	27	28	29
29	30	31	32		28	29	30	31	32	1

图 4-4 E 扩展映射

由图 4-4 的右侧可见，E-运算也可称为加边运算。

记 $E(x_1x_2\cdots x_{32}) = x_{32}x_1\cdots x_{31}x_{32}x_1 = x_{e(1)}x_{e(2)}\cdots x_{e(i)}\cdots x_{e(46)}x_{e(47)}x_{e(48)}$，

那么：$e(i+1)=\left(32+i-2\left[\frac{i}{6}\right]-1\right) \bmod 32+1$ ，$i=0,1,\cdots,47$。

（3）S-盒运算

S-盒运算由 8 个 S-盒函数构成，即：

$S(x_1x_2\cdots x_{48}) = S_1(x_1\cdots x_6)S_2(x_7\cdots x_{12})\cdots S_8(x_{43}\cdots x_{48})$。

每一个 S-盒都是 6 比特的输入，4 比特的输出。$S_i(h_1h_2h_3h_4h_5h_6)$ 的值就是对应表 s_i 中 h_1h_6 行，$h_2h_3h_4h_5$ 列处的数值，记为：

$S_i(h_1h_2h_3h_4h_5h_6) = s_i\left((h_1h_6)_2,(h_2h_3h_4h_5)_2\right)$。

表 4-1 S-Box 表

s_1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

（续）

s_2	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

s_3	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

s_4	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

s_5	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

s_6	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

s_7	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

s_8	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

例如：$S_1(110101)=s_1(11,1010)=s_1(3,10)=3$

（4）P-置换

P-置换是对 8 个 S-盒的输出进行变换，可以扩散单个 S-盒的效果。变换如图 4-5：

1	2	3	4		15	6	19	20
5	6	7	8		28	11	27	16
9	10	11	12		0	14	22	25
13	14	15	16	$\overset{P}{\Rightarrow}$	4	17	30	9
17	18	19	20		1	7	23	13
21	22	23	24		31	26	2	8
25	26	27	28		18	12	29	5
29	30	31	32		21	10	3	24

图 4-5　P 置换

2. DES 结构

设明文为 x，$x_0=L_0R_0=IP(x)$，L_0 表示 x_0 的左边 32 比特，R_0 表示 x_0 的右边 32 比特。记第 i 轮变换为 T_i，轮密钥为 k_i，$i=1,2,\cdots,16$，左右交换变换为 ρ，

则：$DES=IP^{-1}\rho T_{16}T_{15}\cdots T_2T_1IP$。

$$x_0=L_0R_0,$$
$$x_i=T_i(x_{i-1}),\quad x_i=L_iR_i,$$
$$L_i=R_{i-1},R_i=L_{i-1}\oplus f(R_{i-1},k_i),i=1,2,\cdots,16。$$
$$c=\rho(L_{16},R_{16})=(R_{16},L_{16})。$$

DES 加密过程如图 4-6 所示：

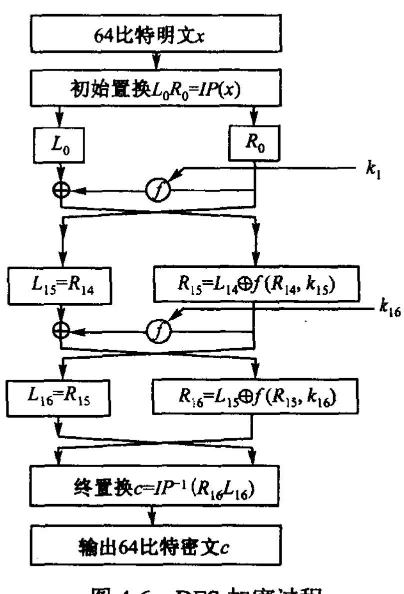


图 4-6　DES 加密过程

其中 f 函数为： $f(R,k)=P\left(S\left(k\oplus E(R)\right)\right)$，如图 4-7 所示：

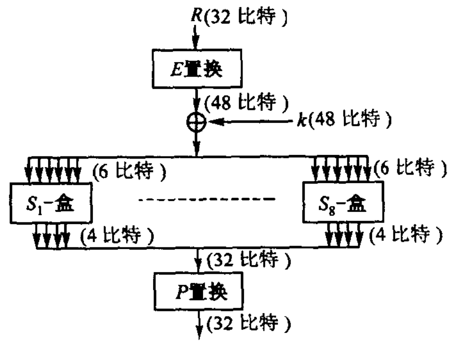


图 4-7　DES 的 f 函数

3．解密算法

DES 解密算法与加密算法完全一样，所不同的就是解密子密钥与加密子密钥的使用顺序相反，即：解密子密钥为：$k_{16},k_{15},\cdots,k_2,k_1$。

$c=IP^{-1}\rho T_{16}T_{15}\cdots T_1 IP(x)$，则解密过程为：

$x=IP^{-1}\rho T_1T_2\cdots T_{15}T_{16}IP(c)$。

下面来证明这一点，记轮变换为 T，则：

$$T(L,R)=\left(R,L\oplus f(R,k)\right)$$

$$\begin{aligned}T\rho T(L,R)&=T\rho\left(R,L\oplus f(R,k)\right)=T\left(L\oplus f(R,k),R\right)\\&=\left(R,\left(L\oplus f(R,k)\right)\oplus f(R,k)\right)\\&=(R,L)\\&=\rho(L,R)\end{aligned}$$

所以 $T\rho T=\rho$。

于是：

$$\begin{aligned}DES^{-1}\left(DES(x)\right)&=IP^{-1}\rho T_1T_2\cdots T_{15}T_{16}IP\left(IP^{-1}\rho T_{16}T_{15}\cdots T_1IP(x)\right)\\&=IP^{-1}\rho T_1T_2\cdots T_{15}T_{16}\rho T_{16}T_{15}\cdots T_1IP(x)\\&=IP^{-1}\rho T_1T_2\cdots T_{15}\rho T_{15}\cdots T_1IP(x)\\&=\cdots\\&=IP^{-1}\rho\rho IP(x)\\&=x\end{aligned}$$

4．子密钥生成器

DES 密钥是一个 64 比特的分组，但是其中 8 个比特是用于奇偶校验的，所以密钥有效位只有 56 比特。由此 56 比特，经过密钥生成器生成 16 轮子密钥。密钥生成器如图 4-8 所示：

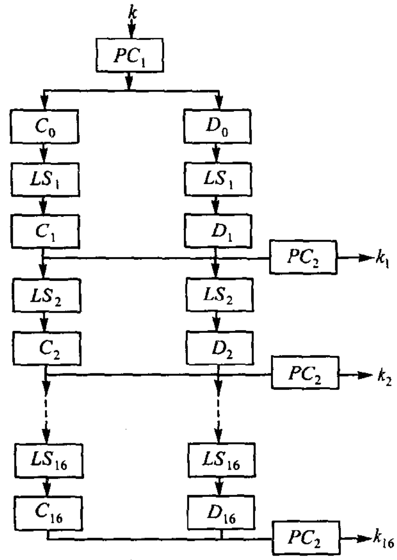


图 4-8　DES 密钥生成器

主要有 PC_1，PC_2 和循环左移操作，其中 C_i，D_i 分别表示左边 28 比特和右边 28 比特。

PC_1 置换如图 4-9 所示：

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	32	33	34	35
36	37	38	39	40	41	42
43	44	45	46	47	48	49
50	51	52	53	54	55	56

$\overset{PC_1}{\Longrightarrow}$

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

图 4-9　PC_1 置换

循环左移 LS_i 见表 4-2。

表 4-2　循环左移变换 LS_i 参数表

轮 i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
LS_i	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

用迭代式表示为：

$$C_0D_0 = PC_1(k)$$

$$C_i = LS_i(C_{i-1}),\quad D_i = LS_i(D_{i-1}),\quad k_i = PC_2(C_iD_i),\quad i = 1, 2, \cdots, 16$$

4.1.3 DES 实例

明文信息为：Message=a m e r i c a n。

其 ASCAII 码的二进制形式为：

Message-Bits=01100001 01101101 01100101 01110010 01101001 01100011 01100001 01101110。

设密钥为：Key= “c r y p t o g r”。

其 ASCAII 码的二进制形式为：

Key−Bits=01100011 01110010 01111001 01110000 01110100 01101111 01100111 01110010。

1．密钥生成过程

密钥生成过程如下：

$PC_{-1}(\text{Key}-\text{Bits})=000000001111111111111111100111100011011100000001001001110$,

$C_0=0000000011111111111111111001$, $D_0=0000000011111111111111111001$,

$C_1=LS_1(C_0)=0000000111111111111111110010$, $D_1=LS_1(D_0)=0000000111111111111111110010$,

$k_1=PC_{-2}(C_1D_1)=111100001011111001101110110111011101111000111100$,

$C_2=LS_2(C_1)=0000001111111111111111100100$, $D_2=LS_2(D_1)=0000001111111111111111100100$,

$k_2=PC_{-2}(C_2D_2)=111000001011111011110110100110011101111000111100$,

$C_3=LS_3(C_2)=0000111111111111111110010000$, $D_3=LS_3(D_2)=0000111111111111111110010000$,

$k_3=PC_{-2}(C_3D_3)=111101001111011001110110110110010111111010110100$,

$C_4=LS_4(C_3)=0011111111111111111001000000$, $D_4=LS_4(D_3)=0011111111111111111001000000$,

$k_4=PC_{-2}(C_4D_4)=111001101101011101110010101110010110101010111101$,

$C_5=LS_5(C_4)=1111111111111111100100000000$, $D_5=LS_5(D_4)=1111111111111111100100000000$,

$k_5=PC_{-2}(C_5D_5)=111011101101001101110111101100110111101010010111$,

$C_6=LS\ (C_5)=1111111111111110010000000011$, $D_6=LS_6(D_5)=1111111111111110010000000011$,

$k_6=PC_{-2}(C_6D_6)=101011111101001101011011101101110010001110110111$,

$C_7=LS_7(C_6)=1111111111111001000000001111$, $D_7=LS_7(D_6)=1111111111111001000000001111$,

$k_7=PC_{-2}(C_7D_7)=001011110101001111111011101101110010101111000111$,

$C_8=LS_8(C_7)=1111111111100100000000111111$, $D_8=LS_8(D_7)=1111111111100100000000111111$,

$k_8=PC_{-2}(C_8D_8)=101111110101100111011001011101101010001111010111$,

$C_9=LS_9(C_8)=1111111110010000000011111111$, $D_9=LS_9(D_8)=1111111110010000000011111111$,

$k_9=PC_{-2}(C_9D_9)=000111110101100111011011111101101010000111001111$,

$C_{10}=LS_{10}(C_9)=1111111001000000001111111111$, $D_{10}=LS_{10}(D_9)=1111111001000000001111111111$,

$k_{10}=PC_{-2}(C_{10}D_{10})=001111110110100111011101011001101011011111001111$,

$C_{11}=LS_{11}(C_{10})=1111100100000000111111111111$, $D_{11}=LS_{11}(D_{10})=1111100100000000111111111111$,

$k_{11}=PC_{-2}(C_{11}D_{11})=000111110110110110001101011111101011010111101011$,

$C_{12}=LS_{12}(C_{11})=1110010000000011111111111111$, $D_{12}=LS_{12}(D_{11})=1110010000000011111111111111$,

$k_{12}=PC_{-2}(C_{12}D_{12})=010110110010110110111101011011101101110101101011$,

$C_{13}=LS_{13}(C_{12})=1100100000000111111111111111$, $D_{13}=LS_{13}(D_{12})=1100100000000111111111111111$,

$k_{13}=PC_{-2}(C_{13}D_{13})=110111011010110010101101010011101111110101111010$,

$C_{14}=LS_{14}(C_{13})=0010000000011111111111111111$, $D_{14}=LS_{14}(D_{13})=0010000000011111111111111111$,

$k_{14}=PC_{-2}(C_{14}D_{14})=110100111010111010101110111011011101110101111000$,

$C_{15}=LS_{15}(C_{14})=1000000001111111111111111100$, $D_{15}=LS_{15}(D_{14})=1000000001111111111111111100$,

$k_{15}=PC_{-2}(C_{15}D_{15})=111110001011111010100110110010011101111001111010$,

$C_{16}=LS_{16}(C_{15})=0000000011111111111111111001$, $D_{16}=LS_{16}(D_{15})=0000000011111111111111111001$,

$k_{16}=PC_{-2}(C_{16}D_{16})=111100011011111000100110110110011101111101111000$。

16 轮解密子密钥写成矩阵形式为：

$$\begin{pmatrix} k_1 \\ k_2 \\ k_3 \\ k_4 \\ k_5 \\ k_6 \\ k_7 \\ k_8 \\ k_9 \\ k_{10} \\ k_{11} \\ k_{12} \\ k_{13} \\ k_{14} \\ k_{15} \\ k_{16} \end{pmatrix} = \begin{pmatrix} 111100001011111100110111011011101110111000111100 \\ 111000001011111011110110100110011101111000111100 \\ 111101001111011001110110110110010111111010110100 \\ 111001101101011101110010101110010110101010111101 \\ 111011101101001101110111101100110111101010010111 \\ 101011111101001101011011101101110010001110110111 \\ 001011110101001111111011101101110010101111000111 \\ 101111110101100111011001011101101010001111010111 \\ 000111110101100111011011111101101010000111001111 \\ 001111110110100111011101011001101011011111001111 \\ 000111110110110110001101011111101011010111101011 \\ 010110110010110110111101011011101101110101101011 \\ 110111011010110010101101010011101111110101111010 \\ 110100111010111010101110111011011101110101111000 \\ 111110001011111010100110110010011101111001111010 \\ 111100011011111100010011011011001110111110111000 \end{pmatrix}。$$

2．加密过程

（1）初始置换

$L_0R_0=IP(\text{Message-Bits})$；

第 1 轮：

L_0=11111111000010001000011001110111， R_0=00000000111111111001001010101000，

k_1=111100001011111100110111011011101110111000111100，

$E(R_0)$=000000000001011111111111111001010010101010101010000，

$E(R_0)\oplus k_1$=111100001010100110010001000101111000101101101100，

$S(E(R_0)\oplus k_1)$=01011011100101000010000110101110，

$f(R_0,k_1)=P(S(E(R_0)\oplus k_1))$=01001100000010101111010110100011，

$L_1=R_0$，$R_1=L_0\oplus f(R_0,k_1)$=10110011000000100111001111010100；

第 2 轮：

k_2=111000001011111011110110100110011101111000111100，

$E(R_1)$=010110100110100000000010000111010011111101010101001，

$E(R_1)\oplus k_2$=101110101101011011110010101000111010000010010101，

$S(E(R_1)\oplus k_2)$=10110100101100011010110110110110，

$f(R_1,k_2)=P(S(E(R_1)\oplus k_2))$=10010111100000100010011110111111，

$L_2=R_1$，$R_2=L_1\oplus f(R_1,k_2)$=10010111011111101101101010001011 1；

第 3 轮：

k_3=111101001111011001110110110110010111111010110100，

$E(R_2)$=110010101110101111111011110110101010100010101111，

$E(R_2)\oplus k_3$=001111100001110110001101000000111101011000011011，

$S(E(R_2)\oplus k_3)$=00011101110000000010100001011110，

$f(R_2,k_3)=P(S(E(R_2)\oplus k_3))$=00011010000110110100000110110010，

$L_3=R_2$，$R_3=L_2\oplus f(R_2,k_3)$=10101001000110010011001001100110；

第 4 轮：

k_4=111001101101011101110010101110010110101010111101，

$E(R_3)$=010101010010100011110010100110100100001100001101，

$E(R_3)\oplus k_4$=101100111111111110000000001000110010100110110000，

$S(E(R_3)\oplus k_4)$=00101001011101110111000011010000，

$f(R_3,k_4)=P(S(E(R_3)\oplus k_4))$=10100110010111010101001010000101，

$L_4=R_3$，$R_4=L_3\oplus f(R_3,k_4)$=00110001001000001110011110010010；

第 5 轮：

k_5=111011101101001101110111101100110111101010010111，

$E(R_4)$=000110100010100100000001011100001111110010100100，

$E(R_4)\oplus k_5$=111101001111101001110110110000111000011000110011，

$S(E(R_4)\oplus k_5)$=01101110011011101111000101011100，

$f(R_4,k_5)=P(S(E(R_4)\oplus k_5))$=01101011010111011011001011110100，

$L_5=R_4$，$R_5=L_4\oplus f(R_4,k_5)$=11000010010001001000000010010010；

第 6 轮：

k_6=101011111101001101011011101101110010001110110111，

$E(R_5)$=011000000100001000001001010000000001010010100101，

$E(R_5)\oplus k_6$=110011111001000101010010111101110011011100010010，

$S(E(R_5)\oplus k_6)$=10110000000000100101111001101001，

$f(R_5,k_6)=P(S(E(R_5)\oplus k_6))$=00111000111101000000111000001010，

$L_6=R_5$，$R_6=L_5\oplus f(R_5,k_6)$=00001001110101001110100110011000；

第 7 轮：

k_7=001011110101001111111011101101110010101111000111，

$E(R_6)$=000001010011111010101001011101010011110011110000，

$E(R_6)\oplus k_7$=001010100110110101010010110000100001011100110111，

$S(E(R_6)\oplus k_7)$=11111011111000101111010001100000，

$f(R_6,k_7)=P(S(E(R_6)\oplus k_7))$=01100001110111011100011110001110，

$L_7=R_6$，$R_7=L_6\oplus f(R_6,k_7)$=10100011100110010100011100011100，

第 8 轮：

k_8=101111110101100111011001011101101010001111010111，

$E(R_7)$=010100000111110011110010101000001110100011111001，

$E(R_7)\oplus k_8$=111011110010010100101011110101100100101100101110，

$S(E(R_7)\oplus k_8)$=00001000110000010000111101110010，

$f(R_7,k_8)=P(S(E(R_7)\oplus k_8))$=10010010001110110010010100001000，

$L_8=R_7$，$R_8=L_7\oplus f(R_7,k_8)$=10011011111011111100110010010000；

第 9 轮：

k_9=000111110101100111011011111101101010000111001111，

$E(R_8)$=010011110111111101011111111001011001010010100001，

$E(R_8)\oplus k_9$=010100000010011010000100000100110011010101101110，

$S(E(R_8)\oplus k_9)$=01100001010011100100111001010010，

$f(R_8,k_9)=P(S(E(R_8)\oplus k_9))$=00010010011101111101001001001000，

$L_9=R_8$，$R_9=L_8\oplus f(R_8,k_9)$=10110001111011101001010101010100；

第 10 轮：

k_{10}=001111110110100111011101011001101011011111001111，

$E(R_9)$=010110100011111101011101010010101010101010101001，

$E(R_9)\oplus k_{10}$=011001010101011010000000001011000001110101100110，

$S(E(R_9)\oplus k_{10})$=10010001010001110111101000000001，

$f(R_9,k_{10})=P(S(E(R_9)\oplus k_{10}))$=10110000111001010101100010000010，

$L_{10}=R_9$，$R_{10}=L_9\oplus f(R_9,k_{10})$=00101011000010101001010000010010；

第 11 轮：

k_{11}=000111110110110110001101011111101011010111101011，

$E(R_{10})$=000101010110100001010101010010101000000010100100，

$E(R_{10})\oplus k_{11}$=000010100000010111011000001101000011010101001111，

$S(E(R_{10})\oplus k_{11})$=01000000111010111101111101010100，

$f(R_{10},k_{11})=P(S(E(R_{10})\oplus k_{11}))$=10110011011101011010000101101100，

$L_{11}=R_{10}$，$R_{11}=L_{10}\oplus f(R_{10},k_{11})$=00000010100110110011010000111000；

第 12 轮：

k_{12}=010110110010110110111101011011101101110101101011，

$E(R_{11})$=000000000101010011110110100110101000000111110000，

$E(R_{11})\oplus k_{12}$=010110110111100101001011111101000101110010011011，

$S(E(R_{11})\oplus k_{12})$=11001100110111110101010011111110，

$f(R_{11},k_{12})=P(S(E(R_{11})\oplus k_{12}))$=10101110110111111001010101111001，

$L_{12}=R_{11}$，$R_{12}=L_{11}\oplus f(R_{11},k_{12})$=10000101110101010000000101101011；

第 13 轮：

k_{13}=110111011010110010101101010011101111110101111010，

$E(R_{12})$=1100000010111110101010101000000000010101101010111，

$E(R_{12})\oplus k_{13}$=000111010001001000000111111001110110101011000101101，

$S(E(R_{12})\oplus k_{13})$=01001100011001011111111101011000，

$f(R_{12},k_{13})=P(S(E(R_{12})\oplus k_{13}))$=10111011001111011011000010011100，

$L_{13}=R_{12}$，$R_{13}=L_{12}\oplus f(R_{12},k_{13})$=10111001101001101000010010100100；

第 14 轮：

k_{14}=110100111010111010101110111011011101110101111000，

$E(R_{13})$=010111110011110100001101010000001001010100001001，

$E(R_{13})\oplus k_{14}$=100011001001001110100011101011010100100001110001,

$S(E(R_{13})\oplus k_{14})$=11001111010111111110001101101111,

$f(R_{13},k_{14})=P(S(E(R_{13})\oplus k_{14}))$=11001101111111111111110011110000,

$L_{14}=R_{13}$, $R_{14}=L_{13}\oplus f(R_{13},k_{14})$=01001000001010101111110110011011;

第 15 轮:

k_{15}=111110001011111010100110110010011101111001111010,

$E(R_{14})$=101001010000000101010101011111111011110011110110,

$E(R_{14})\oplus k_{15}$=010111011011111111110011101101100110001010001100,

$S(E(R_{14})\oplus k_{15})$=10111001110001000010010100001011,

$f(R_{14},k_{15})=P(S(E(R_{14})\oplus k_{15}))$=00001000100010110111101110001010,

$L_{15}=R_{14}$, $R_{15}=L_{14}\oplus f(R_{14},k_{15})$=10110001001011011111111100101110;

第 16 轮:

k_{16}=111100011011111000100110110110011101111101111000,

$E(R_{15})$=010110100010100101011011111111111110100101011101,

$E(R_{15})\oplus k_{16}$=101010111001011101111101001001100011011000100101,

$S(E(R_{15})\oplus k_{16})$=01100000111100100100001101011110,

$f(R_{15},k_{16})=P(S(E(R_{15})\oplus k_{16}))$=00001110011101111010001100100100,

$L_{16}=R_{15}$, $R_{16}=L_{15}\oplus f(R_{15},k_{16})$=01000110010111010101111010111111;

（2）输出置换（初始置换的逆置换）

$Cipher=IP^{-1}(R_{16}(L_{16})$,

$Cipher$=1011100101001111011111110011111110011101101010110101110010001001。

3. 解密过程

解密子密钥与加密子密钥的顺序相反，解密过程与加密过程完全一样:

L_0=01000110010111010101111010111111， R_0=10110001001011011111111100101110。

$L_1=R_0$, $R_1=L_0\oplus f(R_0,k_1)$=01001000001010101111110110011011,

$L_2=R_1$, $R_2=L_1\oplus f(R_1,k_2)$=10111001101001101000010010100100,

$L_3=R_2$, $R_3=L_2\oplus f(R_2,k_3)$=10000101110101010000000101101011,

$L_4=R_3$, $R_4=L_3\oplus f(R_3,k_4)$=00000010100110110011010000111000,

$L_5=R_4$, $R_5=L_4\oplus f(R_4,k_5)$=00101011000010101001010000010010,

$L_6=R_5$, $R_6=L_5\oplus f(R_5,k_6)$=10110001111011101001010101010100,

$L_7=R_6$, $R_7=L_6\oplus f(R_6,k_7)$=10011011111011111100110010010000,

$L_8=R_7$, $R_8=L_7\oplus f(R_7,k_8)$=10100011100110010100011100011100,

$L_9=R_8$, $R_9=L_8\oplus f(R_8,k_9)$=00001001110101001110100110011000,

$L_{10}=R_9$, $R_{10}=L_9\oplus f(R_9,k_{10})$=11000010010001001000000010010010,

$L_{11}=R_{10}$, $R_{11}=L_{10}\oplus f(R_{10},k_{11})$=00110001001000001110011110010010,

$L_{12}=R_{11}$, $R_{12}=L_{11}\oplus f(R_{11},k_{12})$=10101001000110010011001001100110,

$L_{13}=R_{12}$, $R_{13}=L_{12}\oplus f(R_{12},k_{13})$=10010111011111011011010100010111,

$L_{14}=R_{13}$, $R_{14}=L_{13}\oplus f(R_{13},k_{14})$=10110011000000100111001111010100,

$L_{15}=R_{14}$，$R_{15}=L_{14}\oplus f(R_{14},k_{15})$=00000000111111111001001010101000，

$L_{16}=R_{15}$，$R_{16}=L_{15}\oplus f(R_{15},k_{16})$=11111111000010001000011001110111，
Plaintext=0110000101101101011001010111001001101001011000110110000101101110。

4．雪崩效应验证：

已知：*Key*=0110001101110010011110010111000001110100011011110110011101110010，

明文：*Plainttext*1=0110000101101101011001010111001001101001011000110110000101101110，

则密文：*Cipher*1=1011100101001111011111110011111110011101101010110101110010001001。

修改 Plainttext1 的第 1 个比特：

*Plainttext*2=1110000101101101011001010111001001101001011000110110000101101110

得到密文：

*Cipher*2=0010110001111000011001010100100110111010011011111110100000111010

可见密文 *Cipher*1，*Cipher*2 的异或为：

1001010100110111000110100111011000100111110001001011010010110011

二者相差 33 个比特；

同样，如果修改 Plainttext1 的第 2 个比特：

*Plainttext*3=0010000101101101011001010111001001101001011000110110000101101110

则密文：

*Cipher*3=1011111110100100001001110101011100111011010011100010111011010111

可见 *Cipher*1，*Cipher*3 也相差 32 个比特；

同样固定明文，修改密钥的一个比特，也将引起密文的至少一半以上比特的变化。

密码学把这种密文对明文、密钥的敏感性称为雪崩效应。

4.1.4 DES 的安全性

DES 的安全性一开始就受到怀疑，主要表现在以下几点：

（1）关于 *S*-盒的设计准则

这个问题涉及美国国家安全局（NSA），他们修改了 IBM 的 *S*-盒。虽然修改后的 *S*-盒满足 IBM 原先关于 *S*-盒的设计原则，但是，人们仍因此而怀疑 NSA 在 *S*-盒中嵌入了陷门，使得 NSA 借助于这个陷门及 56 比特的短密钥可以解密 DES。

在 1990 年以前，S-盒的设计准则一直没有公布，直到差分密码分析方法发表后才公布，公布的 *S*-盒设计准则是：

*S*1：*S*-盒的每一行是 0~15 的一个排列；

*S*2：没有一个 *S*-盒接近其输入的线性函数；

*S*3：如果固定输入的最左边，最右边的位固定，变换中间 4 位，每个可能的 4 位输出只能得到一次。

*S*4：如果两个输入仅中间 2 位不同，则输出至少有 2 位不同；

*S*5：如果两个输入仅差 1 位，则输出至少有 2 位不同；

*S*6：如果两个输入前 2 位不同，后 2 位已知，则输出不同；

*S*7：具有非零，相同差分的 32 对输入中，至多有 8 对具有相同的输出差分；

P 函数是为了混合 8 个 *S*-盒的输出，使各 *S*-盒之间相互影响，增强扩散性，它也有相应的准则。

（2）关于 56 比特的短密钥

IBM 最初提交的方案中密钥是 112 比特，但是，公布的 DES 却是 56 比特，坚持使用 56 比特短密钥是 NSA 的意见。这就使得人们更加相信 DES 的陷门之说。

56 比特的短密钥确实太短，下面两个例子很有说服力。

分布式穷举攻击：1997 年 1 月 28 日，美国 RSA 数据安全公司在 Internet 上开展了一项“秘密密钥挑战”的竞赛，悬赏一万美元，破解一段 DES 密文。计划公布后，得到了许多网络用户的积极相应。科罗拉多州的程序员 R.Verser 设计了一个可以通过互联网分段运行的密钥搜索程序，组织了一个称为 DESCHALL 的搜索行动，成千上万的的志愿者加入到计划中。第 96 天，即竞赛公布后的第 140 天，1997 年 6 月 17 日晚上 10 点 39 分，美国盐湖城 Inetz 公司职员 M.Sanders 成功地找到了密钥，解密出明文：“Strong cryptography makes the world a safer place”(高强度密码技术使世界更安全)。

专用搜索机：1998 年 5 月，美国电子边境基金会（EFF, Electronic Frontier Foundation）宣布，他们用一台价值 20 万美元的计算机改装成专用设备，56 小时就破译了 DES。这样，更加直接地说明了 56 比特密钥太短了，彻底宣布了 DES 的终结。

（3）关于差分分析

对 DES 的分析发现，DES 的 *S*-盒具有差分特性，即：其输出的差分（异或）不能遍历 0~2^4-1，不是等可能地在 0~15 之间取值。根据这点，1990 年，以色列密码学家 Eli Biham 和 Adi Shamir 对分组密码提出了差分密码分析方法。这是一种选择明文攻击最为有效的方法。虽然对 16 轮 DES 没有攻破，但是，如果迭代的轮数降低，则 DES 可被成功地攻破。例如，8 轮 DES 在一台 PC 机上只需要 2 分钟即可被攻破。差分分析正是是利用了 DES 的 *S*-盒输出的差分不能均匀遍历 0~15 这一特点。

（4）DES 的对称性

DES 算法具有对称：如果用 $\overline{}$ 表示按位取补，则：$\overline{DES_k(x)} = DES_{\bar{k}}(x)$。

这种对称性，使穷举攻击密钥搜索量可以减少一半。

（5）弱密钥和半弱密钥

如果对于初始密钥 k，生成的子密钥有 $k_1 = k_2 = \cdots = k_{16}$，则称 k 是弱密钥。显然，$DES_k(DES_k(x)) = x$；如果有一对密钥 k_1，k_2，使得 $DES_{k_2}(DES_{k_1}(x)) = DES_{k_1}(DES_{k_2}(x)) = x$，则称 k_1，k_2 是（互逆的）半密钥。

DES 弱密钥有 4 个（十六进制表示）：

01	01	01	01	01	01	01	01
1F	1F	1F	1F	1F	1F	1F	1F
E0	E0	E0	E0	E0	E0	E0	E0
FE	FE	FE	FE	FE	FE	FE	FE

DES 半密钥有 12 个（十六进制表示）：

01	FE	01	FE	01	FE	01	FE
FE	01	FE	01	FE	01	FE	01
1F	E0	1F	E0	1F	E0	1F	E0
E0	1F	E0	1F	E0	1F	E0	1F
01	E0	01	E0	01	E0	01	E0

E0	01	E0	01	E0	01	E0	01
1F	FE	1F	FE	1F	FE	1F	FE
FE	1F	FE	1F	FE	1F	FE	1F
01	1F	01	1F	01	1F	01	1F
1F	01	1F	01	1F	01	1F	01
E0	FE	E0	FE	E0	FE	E0	FE
FE	E0	FE	E0	FE	E0	FE	E0

更一般地，可以定义四分之一弱密钥等。

（6）代数性质

定义 4-1 设 E 是一种加密算法，如果对于任意密钥 k_1，k_2，都存在密钥 k_3，使得对于任意明文 x 都有：$E_{k_2}(E_{k_1}(x))=E_{k_3}(x)$，则称加密算法 E 是（二重）闭合的（closed），也称密钥 k_1，k_2 是闭合的。

定义 4-2 如果对于任意密钥 k_1，k_2，k_3，都存在密钥 k_4，使得对于任意明文 x 都有：$E_{k_3}(E_{k_2}(E_{k_1}(x)))=E_{k_4}(x)$，则称加密算法 E 是三重封合的，称密钥 k_1，k_2，k_3，是闭合的。

类似地可以定义加密算法的多重闭合性及多个密钥的闭合性。

显然，对闭合的加密算法企图通过二重加密来增强体制的安全性（特别是关于穷举攻击的安全性）是无效的，同样，对于三重闭合的加密算法用三重加密来增强安全性也是无效的。

易知，一个二重闭合算法一定也是三重闭合的，反之则不然。

定理 4-1 设 (P,C,E,D,K) 是一个密码体制，如果加密算法 E 是闭合的，且 $C=P$，$D=E$，则 $\{E_k \mid k\in K\}$ 在复合加密运算下是一个群。

证明 对 $\forall k_1,k_2\in K$，及 $\forall x\in P$，定义 $(E_{k_2}\cdot E_{k_1})(x)=E_{k_2}(E_{k_1}(x))$，则由 $\{E_k \mid k\in K\}$ 闭合性知，存在 $k_3\in K$，使得 $E_{k_2}\cdot E_{k_1}=E_{k_3}$。即：复合加密运算“ •”是良定的。

进一步：$((E_{k_1}E_{k_2})E_{k_3})(x)=(E_{k_1}E_{k_2})E_{k_3}(x)=E_{k_1}(E_{k_2}(E_{k_3}(x)))$，

$=E_{k_1}((E_{k_2}E_{k_3})(x))=E_{k_1}(E_{k_2}E_{k_3})(x)$，

所以复合加密运算满足集合律。下面考虑单位元的存在性。

取一个加密密钥 $k\in K$，由于 E 是加密算法，所以它有解密密钥 $k'\in K$，满足 $D_{k'}E_k=I$（I 表示恒等映射），再由条件 $D=E$ 得：$E_{k'}E_k=I$。

再由 E 的闭合性知，对 k,k'，必存在 $k_0\in K$，使得 $E_{k'}\cdot E_k=E_{k_0}$，当然 $E_{k_0}(x)=x$。易见，这个 E_{k_0} 就是所求的群单位元：对 $\forall k\in K$，$E_{k_0}\cdot E_k=E_k\cdot E_{k_0}=E_k$。所以，$\{E_k \mid k\in K\}$ 是一个群。

但一般地，$\{E_k \mid k\in K\}$ 不是可换群。

1992 年，K.W.Campbell，M.J.Wiener 证明了 DES 不是一个群。

正因以上结论，二重 DES，三重 DES 有一定的价值，其穷举攻击的安全性虽有所改进，然而并非 2 倍，3 倍的密钥量那么坚强。

在这里，我们还注意到迭代轮数的问题。DES 虽不是一个群，但仍不能排除某些密钥的闭合性。例如：弱密钥、半弱密钥。对于轮变换，某种意义上来说，它也相当于一种加密

算法，从而也可能有闭合性，群结构等性质，也就是说：并不是迭代轮数越多越好，也许 16 轮是最好的选择。

（7）DES 变形

为了提高 DES 的安全性和各种需求，人们设计开发了 DES 的多种变形版本，主要有：

多重 DES：由于 DES 不是一个群，所以，人们采用多重复合来扩展 DES 的密钥空间，比较普遍的是 3DES：$3DES_{(k_2 k_1)} = DES_{k_1} DES_{k_2}^{-1} DES_{k_1}$，其容量为 2^{112}。

独立子密钥 *des*：应用 16 个独立密钥代替轮变换中的子密钥，扩展密钥空间容量为 2^{768}。即 $des = IP^{-1} \rho T_{k_{16}} T_{k_{15}} \cdots T_{k_2} T_{k_1} IP$。

DES S-盒变异：通过替换或改变 DES 的 S-盒来实现 DES，可以消除 DES 陷门的可能。但是，Biham 和 Shamir 证明，现行 DES 的 S-盒设计对抗差分分析是最优的。

DESX：它是 RSA 数据安全公司提出的 DES 的一种变形，它除了 56 比特 DES 密钥外，还有 64 比特随机密钥，在 DES 第一轮之前与明文异或，再将这 64 比特密钥通过一个 Hash 函数变为 120 比特 DES 密钥，并与最后一轮的输出作异或。其目的是掩盖其输入、输出。DESX 比 DES 具有更强的抗穷举攻击和抗差分、线性分析等能力。

DES 的变形还有 GDES，RDES，s^nDES，CRYPT（3）等等。

4.1.5 DES 工作方式

FIPS PUB74，FIPS PUB81 中给出了 DES 的四种工作方式，这也是一般分组密码的工作方式。

1．电子密码本方式

电子密码本方式（ECB, electronic codebook）是一种最直接、简单的方式，明文以 64 比特分组逐组使用同一个密钥加密。显然，如果明文具有重复成分，其长度是 64 比特的倍数，那么将会得到相同的密文分组。这将为密码分析者提供重要信息。对 ECB 方式的一个有效攻击就是重放攻击，即：攻击者在不知道密钥的情况下通过重放可以修改密文。总之，这种方式是一种不安全的方式。

ECB 加密过程如图 4-10 所示：

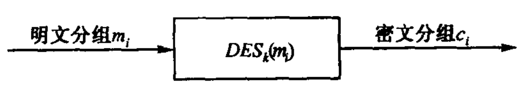


图 4-10 ECB 加密模式

解密过程如图 4-11 所示：

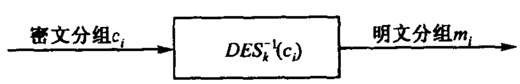


图 4-11 ECB 解密模式

即：$c_i = DES_k(m_i)$，$m_i = DES_k^{-1}(c_i)$，$i = 1, 2, \cdots$。

2. 密码分组链接方式

密码分组链接方式（CBC, cipher block chaining）很好地弥补了 ECB 的缺陷。它用前一个分组的密文与当前分组明文作异或后再加密，从而消除了相同明文分组加密后密文相同的现象。但是，很显然，这种方式具有错误传播的特点。也就是说，密文中 1 个比特的错误将会影响当前密文分组及以后的所有密文分组的解密。

CBC 加密过程如图 4-12 所示：

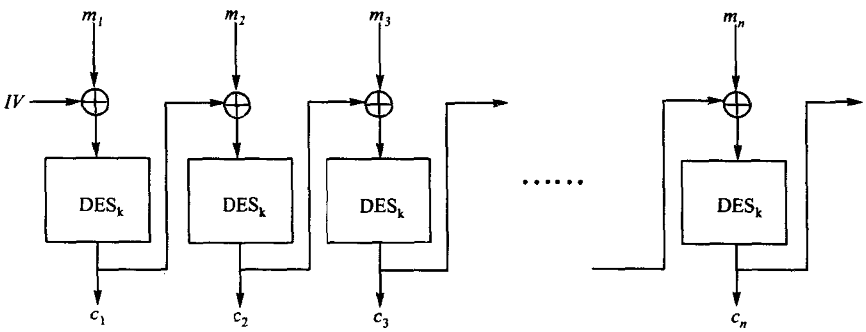


图 4-12　CBC 加密模式

CBC 解密过程如图 4-13 所示：

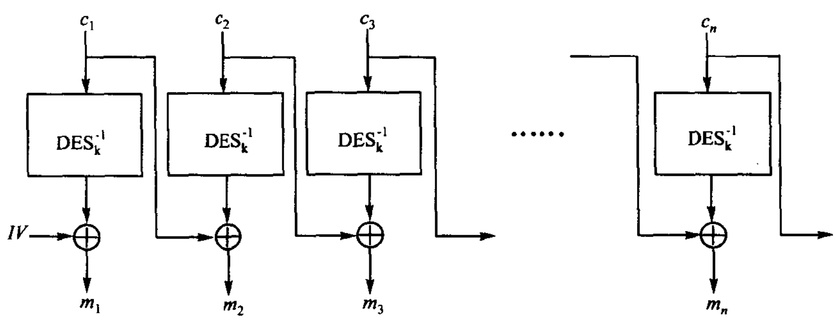


图 4-13　CBC 解密模式

即：$c_0 = IV$，是初始向量。

$c_i = DES_k(m_i \oplus c_{i-1})$ ，　$m_i = c_{i-1} \oplus DES_k^{-1}(c_i)$，$i = 1,2,\cdots$。

3. 密码反馈方式

密码反馈方式（CFB, Cipher-FeedBack）用当前明文分组与前一分组的的加密后的密文作异或。仍需要一个初始向量 IV。

CFB 加密过程如图 4-14 所示：

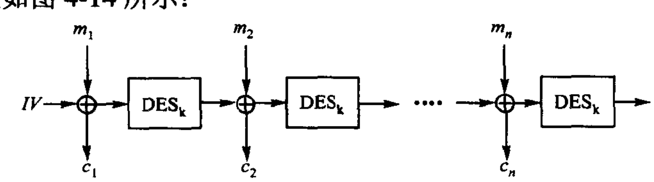


图 4-14　CFB 加密模式

CFB 解密过程如图 4-15 所示：

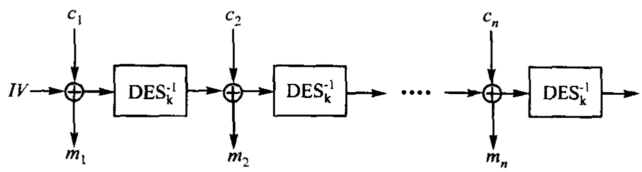


图 4-15　CFB 解密模式

即：$c_0 = IV$，

$c_i = m_i \oplus DES_k(c_{i-1})$　，　$m_i = c_i \oplus DES_k(c_{i-1})$　，　$i = 1,2,\cdots$。

4．输出反馈方式

输出反馈（Out put-Feed Back，OFB）是用一个初始向量逐次加密再分别与相应分组作异或。

OFB 加密过程如图 4-16 所示：

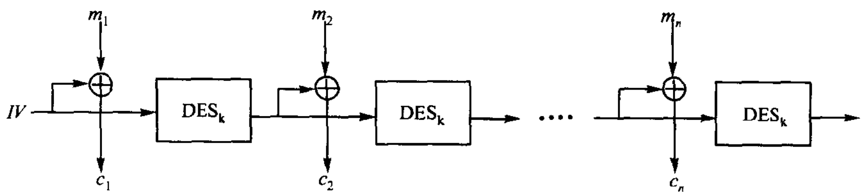


图 4-16　OFB 加密模式

OFB 解密过程如图 4-17 所示：

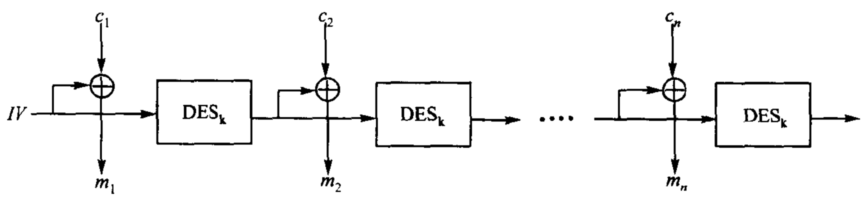


图 4-17　OFB 解密模式

即：$s_0 = IV$，

$c_i = m_i \oplus s_i$　，　$s_i = DES_k(s_{i-1})$　；$m_i = c_i \oplus s_i$　，　$s_i = DES_k(s_{i-1})$　。

分组密码体制除以上四种工作模式以外，还有其他工作方式，它们都有各自的优缺点。

4.2　国际数据加密算法 IDEA

DES 自发表以后，一直受到怀疑和批评，而各种替代算法不断被提出。1990 年，我国学者来学嘉和瑞士著名密码学家马赛（James Massey）提出了 IDEA。其第一版称为 PES（Proposed Encryption Standard，推荐加密标准），第二版称为 IPES（Improved Proposed Encryption Standard，改进型推荐加密标准），可以抗击差分分析，1992 年 IPES 改名为 IDEA（International Data Encryption Algorithm，国际数据加密算法）。其目的就是要替代 DES 成为真正的国际数据加密标准。IDEA 取得了较大的成功，被广泛地使用于 PGP 等著名保密系

统，也取得了各种专利。

4.2.1 IDEA 加密算法

IDEA 是一个密钥 128 比特，明文分组为 64 比特的分组密码体制。

64 比特分组被分成 4 个 16 比特的子块：X_1, X_2, X_3, X_4，作为第一轮的输入。

每一轮中，将四个输入子块与六个 16 比特子密钥分别作模 2^{16} 的加法、模 $2^{16}+1$ 的乘法、异或等操作，最后得到四个输出，作为下一轮的输入。

这样，共进行 8 轮，最后用四个子密钥作输出变换。IDEA 整体结构如图 4-18 所示.

8 轮及输出变换共需要 52 个子密钥，由子密钥生成器生成。下面详细分解 IDEA 轮变换和其密钥生成过程。

1）轮变换如图 4-19 所示。

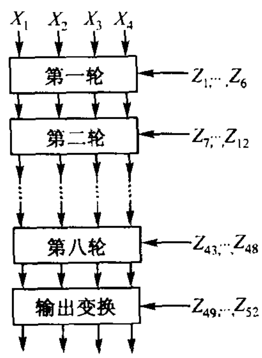


图 4-18 IDEA 加密流程

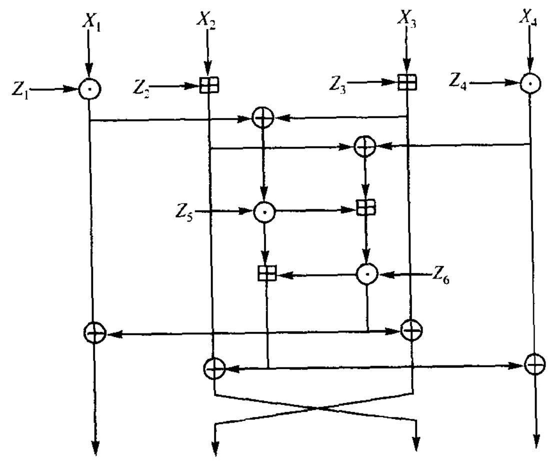


图 4-19 IDEA 轮变换

2）输出变换如图 4-20 所示。

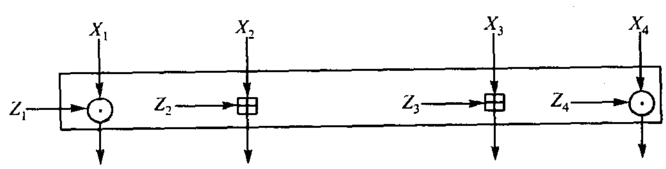


图 4-20 IDEA 输出变换

其中⊙是模 $2^{16}+1$ 的乘法，⊞是模 2^{16} 的加法，即：

$A \odot B=(AB) \bmod 2^{16}+1$，$A \boxplus B=(A+B) \bmod 2^{16}$。

记轮变换为 T，输入子块为 X_1, X_2, X_3, X_4，子密钥为 $Z_1, Z_2, Z_3, Z_4, Z_5, Z_6$。

轮变换可以分解为两个变换，分别为 H-变换和非线性变换 S-变换。

H-变换如图 4-21 所示：

$H(X_1,\ldots,X_4;\ Z_1,\ldots,Z_4)=(H_1,\ldots,H_4)$

$H_1=X_1 \odot Z_1$，$H_2=X_2 \boxplus Z_2$，$H_3=X_3 \boxplus Z_3$，$H_4=X_4 \odot Z_4$。

S-变换的核心变换是非线性乘/加运算，称为 S-Box，见图 4-22，运算表达式为：$S_R=((Y_1\odot Z_5)\boxplus Y_2)\odot Z_6$，$S_L=(Y_1\odot Z_5)\boxplus S_R$，见图 4-23。

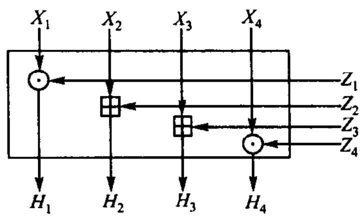


图 4-21　IDEA 的 H 变换

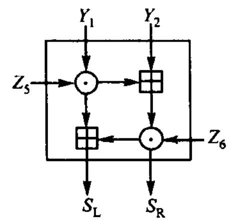


图 4-22　IDEA 的 S-Box

轮变换（图 4-24）可结构化为：

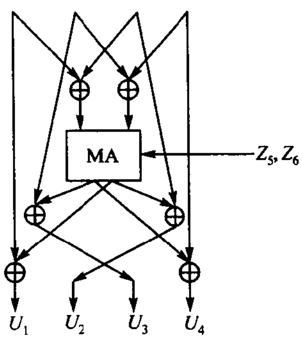


图 4-23　IDEA 的 S-box 变换

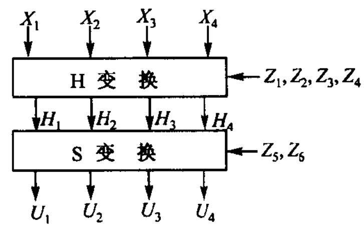


图 4-24　IDEA 轮变换

$T(X_1,\ldots,X_4;\ Z_1,\ldots,Z_6)=(U_1,\ldots,U_4)$，$H(X_1,\ldots,X_4;\ Z_1,\ldots,Z_4)=(H_1,\ldots,H_4)$；

$U_1=H_1\oplus S_R(H_1\oplus H_3,\ H_2\oplus H_4;\ Z_5,Z_6)$，

$U_2=H_3\oplus S_R(H_1\oplus H_3,\ H_2\oplus H_4;\ Z_5,Z_6)$，

$U_3=H_2\oplus S_L(H_1\oplus H_3,\ H_2\oplus H_4;\ Z_5,Z_6)$，

$U_4=H_4\oplus S_L(H_1\oplus H_3,\ H_2\oplus H_4;\ Z_5,Z_6)$。

（3）输出变换由一个换位变换和 H 变换构成

换位变换记为 ρ： $\rho(A,B,C,D)=(A,C,B,D)$。

这样，如果将第 i 轮（含最后的输出变换）的 H-变换、S-变换分别记为 H_i， S_i，那么：

$IDEA=H_9\rho S_8H_8S_7H\ \cdots S_1H_1$。

4.2.2　IDEA 子密钥生成

IDEA 共需要 52 个子密钥，每一个有 16 比特，由 128 比特密钥生成。子密钥将 128 比特分成 8 组，每组 16 比特，得到 $k_1,k_2,\cdots,k_8$；将这 128 比特循环左移 25 位后作 16 比特分组，得到子密钥 $k_9,k_{10},\cdots,k_{16}$；再将这 128 比特循环左移位后作同样的分组得到子密钥 $k_{17},k_{18},\cdots,k_{24}$；如此继续，直至所有子密钥都生成。

第 1、2 轮子密钥如图 4-25 所示：

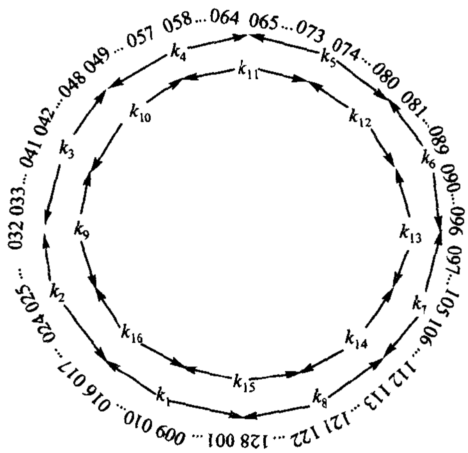


图 4-25　第 1、2 轮子密钥生成示意图

如果记 IDEA 密钥为：$Key = t_1 t_2 \cdots t_{128}$，$t_i = 0,1$，$i = 1,2,\cdots,128$，则其子密钥为：$k_i = t_{(j+1 \bmod 128)} t_{(j+2 \bmod 128)} \cdots t_{(j+16 \bmod 128)}$，其中 $j = (i-1)*16 + (i \bmod 8)*25$。

4.2.3 IDEA 解密

IDEA 解密算法与加密算法具有同样的的结构，将密文分组分成 4 个 16 比特的块作为输入，但是，解密密钥有所不同，加密密钥与解密密钥的关系见表 4-3。

表 4-3　IDEA 加、解密子密钥关系表

轮　数	加密子密钥	解密子密钥
第 1 轮	Z_1 Z_2 Z_3 Z_4 Z_5 Z_6	Z_{49}^{-1} $-Z_{50}$ $-Z_{51}$ Z_{52}^{-1} Z_{47} Z_{48}
第 2 轮	Z_7 Z_8 Z_9 Z_{10} Z_{11} Z_{12}	Z_{43}^{-1} $-Z_{45}$ $-Z_{44}$ Z_{46}^{-1} Z_{41} Z_{42}
第 3 轮	Z_{13} Z_{14} Z_{15} Z_{16} Z_{17} Z_{18}	Z_{37}^{-1} $-Z_{39}$ $-Z_{38}$ Z_{40}^{-1} Z_{35} Z_{36}
第 4 轮	Z_{19} Z_{20} Z_{21} Z_{22} Z_{23} Z_{24}	Z_{31}^{-1} $-Z_{33}$ $-Z_{32}$ Z_{34}^{-1} Z_{29} Z_{30}
第 5 轮	Z_{25} Z_{26} Z_{27} Z_{28} Z_{29} Z_{30}	Z_{25}^{-1} $-Z_{27}$ $-Z_{26}$ Z_{28}^{-1} Z_{23} Z_{24}
第 6 轮	Z_{31} Z_{32} Z_{33} Z_{34} Z_{35} Z_{36}	Z_{19}^{-1} $-Z_{21}$ $-Z_{20}$ Z_{22}^{-1} Z_{17} Z_{18}
第 7 轮	Z_{37} Z_{38} Z_{39} Z_{40} Z_{41} Z_{42}	Z_{13}^{-1} $-Z_{15}$ $-Z_{14}$ Z_{16}^{-1} Z_{11} Z_{12}
第 8 轮	Z_{43} Z_{44} Z_{45} Z_{46} Z_{47} Z_{48}	Z_7^{-1} $-Z_9$ $-Z_8$ Z_{10}^{-1} Z_5 Z_6
输出变换	Z_{49} Z_{50} Z_{51} Z_{52}	Z_1^{-1} $-Z_2$ $-Z_3$ Z_4^{-1}

其中：$-Z_i$ 表示 Z_i 模 2^{16} 的加法逆元，即：$-Z_i + Z_i \equiv 0 \bmod 2^{16}$。

Z_i^{-1} 表示 Z_i 模 $2^{16}+1$ 的乘法逆元，即：$Z_i^{-1} Z_i \equiv 0 \bmod 2^{16}+1$。

如果将第 1,2,…,8 个解密轮及输出解密轮中的 H-变换，S-变换分别记为：$H_{-9}, S_{-8}, H_{-8}, S_{-7}, \cdots, H_{-2}, H_{-1}$，那么，$IDEA^{-1} = H_{-1}\rho S_{-1} H_{-2} S_{-2} H_{-3} \cdots S_{-8} H_{-9}$。

$$IDEA^{-1} \cdot IDEA = H_{-1}\rho S_{-1} H_{-2} S_{-2} H_{-3} \cdots S_{-8} H_{-9} \cdot H_9 \rho S_8 H_8 S_7 H_7 \cdots S_1 H_1$$
$$= H_{-1}\rho\Big(S_{-1}\big(H_{-2}\left(S_{-2} \cdots S_{-8}\left(H_{-9} H_9 \rho\right) S_8 \cdots S_2\right) H_2\big) S_1\Big) H_1 。$$

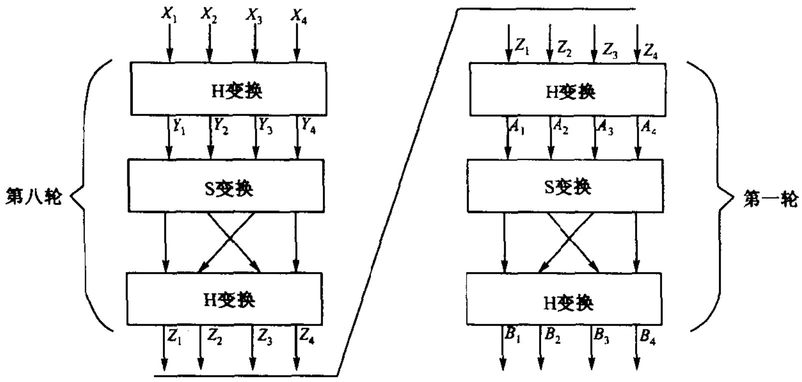


图 4-26 IDEA 解密流程

可见我们只须证明：$H_{-i}H_i = id$ ， $i=1,9$

$S_{-i}\rho S_i = \rho$ ， $H_{-i}\rho H_i = \rho$ ， $i=8,7,\cdots,2$ 。

从解密密钥与加密密钥的关系可见，对于 $i=8,7,\cdots,2$ ，如果 H_i 的密钥是 k_1,k_2,k_3,k_4 ，那么 H_{-i} 的密钥是 $k_1^{-1},-k_3,-k_2,k_4^{-1}$ ，而对 $i=1,9$ ， H_{-i} 的密钥是 $k_1^{-1},-k_2,-k_3,k_4^{-1}$ ；对于 $i=1,2,\cdots,8$ ， S_i 与 S_{-i} 的密钥相同。于是，不失一般性，轮密钥一律用 k_1,k_2,k_3,k_4,k_5,k_6 表示。

（1）首先证明： $H_{-i}H_i = id$ ， $i=1,9$ 。

$H_{-i}(H_i(A_1, A_3, A_3, A_4;\ Z_1,\ldots,Z_4)) = H_{-i}(A_1 \odot k_1,\ A_2 ⊞ k_2,\ A_{32} ⊞ k_3,\ A_4 \odot k_4)$

$= (A_1 \odot k_1 \odot k_1^{-1},\ A_{22} ⊞ k_{22} ⊞ (-k_2),\ A_{32} ⊞ k_{32} ⊞ (-k_3),\ A_4 \odot k_4 \odot k_4^{-1})$

$=(A_1, A_3, A_3, A_4)$。

所以：$H_{-i}\,H_i = id$。

（2）其次证明： $S_{-i}\rho S_i = \rho$ 。

记 $S_i(A_1,A_2,A_3,A_4)=(B_1,B_2,B_3,B_4)$ ，则：

$B_1 = A_1 \oplus S_R(A_1 \oplus A_3, A_2 \oplus A_4; k_1,k_2)$ ，

$B_2 = A_3 \oplus S_R(A_1 \oplus A_3, A_2 \oplus A_4; k_1,k_2)$ ，

$B_3 = A_2 \oplus S_L(A_1 \oplus A_3, A_2 \oplus A_4; k_1,k_2)$ ，

$B_4 = A_4 \oplus S_L(A_1 \oplus A_3, A_2 \oplus A_4; k_1,k_2)$ 。

$S_{-i}\rho S_i(A_1,A_2,A_3,A_4)$

$= S_{-i}\rho(B_1,B_2,B_3,B_4)$

$= S_{-i}(B_1,B_3,B_2,B_4)$

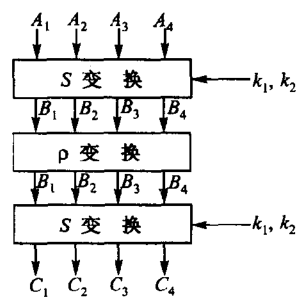


图 4-27 $S_{-i}\rho S_i=\rho$示意图

$B_1 \oplus B_2$

$=(A_1 \oplus S_R(A_1 \oplus A_3, A_2 \oplus A_4;\ k_1,k_2)) \oplus (A_3 \oplus S_R(A_1 \oplus A_3, A_2 \oplus A_4;\ k_1,k_2))$

$= A_1 \oplus A_3$

$B_3 \oplus B_4$

$=(A_2 \oplus S_L(A_1 \oplus A_3, A_2 \oplus A_4;\ k_1,k_2)) \oplus (A_4 \oplus S_L(A_1 \oplus A_3, A_2 \oplus A_4;\ k_1,k_2))$

$= A_2 \oplus A_4$

$$
\begin{aligned}
C_1 &= B_1 \oplus S_R(B_1 \oplus B_2, B_3 \oplus B_4;\ k_1, k_2) \\
&= \left(A_1 \oplus S_R(A_1 \oplus A_3, A_2 \oplus A_4;\ k_1, k_2)\right) \oplus S_R(A_1 \oplus A_3, A_2 \oplus A_4;\ k_1, k_2) \\
&= A_1 \\
C_2 &= B_2 \oplus S_R(B_1 \oplus B_2, B_3 \oplus B_4;\ k_1, k_2) \\
&= \left(A_3 \oplus S_R(A_1 \oplus A_3, A_2 \oplus A_4;\ k_1, k_2)\right) \oplus S_R(A_1 \oplus A_3, A_2 \oplus A_4;\ k_1, k_2) \\
&= A_3 \\
C_3 &= B_3 \oplus S_L(B_1 \oplus B_2, B_3 \oplus B_4;\ k_1, k_2) \\
&= \left(A_2 \oplus S_R(A_1 \oplus A_3, A_2 \oplus A_4;\ k_1, k_2)\right) \oplus S_R(A_1 \oplus A_3, A_2 \oplus A_4;\ k_1, k_2) \\
&= A_2 \\
C_4 &= B_4 \oplus S_L(B_1 \oplus B_2, B_3 \oplus B_4;\ k_1, k_2) \\
&= \left(A_4 \oplus S_R(A_1 \oplus A_3, A_2 \oplus A_4;\ k_1, k_2)\right) \oplus S_R(A_1 \oplus A_3, A_2 \oplus A_4;\ k_1, k_2) \\
&= A_4 \\
S_{-i}\rho S_i(A_1, A_2, A_3, A_4) &= S_{-i}(B_1, B_3, B_2, B_4) \\
&= (C_1, C_2, C_3, C_4) = (A_1, A_3, A_2, A_4) = \rho(A_1, A_2, A_3, A_4)
\end{aligned}
$$

所以，$S_{-i}\rho S_i = \rho$。

4.2.4 IDEA 讨论

IDEA 有 128 比特，密钥空间容量为 $2^{128} \approx 10^{38}$。对于穷举攻击来说目前已经足够了，如果我们把加密记作一次运算，那么搜索完 IDEA 密钥空间需要 10^{13}MPIS 年（1 MPIS 年即每秒运行 100 万次的计算机运行一年）。

IDEA 没有 DES 意义下的弱密钥，能够抗击差分分析和相关分析。

如果有一台每秒能够穷举 DES 密钥空间的设备，即每秒加密 2^{55} 次（DES 具有对称性，所以，空间穷举量减少一半），它要穷举 IDEA 密钥空间则需要 299 万亿年。

IDEA 的基本运算 A 田 B=A+B mod 2^{16}是 16 位无进位的加法运算，所以运算效率高。而 A⊙B=AB mod（2^{16}+1）则是最为耗时的。但是下面的结论可以提高模运算 mod（2^{16}+1）的速度：

定理 4-2 （低—高位算法）

设 $0 < a,b < 2^{16}+1$ 是整数，则：

$$
ab \mod (2^{16}+1) = \begin{cases} \left(ab \mod 2^{16}\right) - \left[ab/2^{16} \right], & \left(ab \mod 2^{16}\right) \geqslant \left[ab/2^{16} \right] \\ \left(ab \mod 2^{16}\right) - \left[ab/2^{16} \right] + 2^{16} + 1, & \left(ab \mod 2^{16}\right) < \left[ab/2^{16} \right] \end{cases},
$$

其中 $\left[ab/2^{16} \right]$ 表示 ab 除以 2^{16} 的商。

证明 对任意整数 $0 < a,b < 2^{16}+1$，存在 q,r，满足：

$ab = q\left(2^{16}+1\right) + r$， $0 \leqslant q < 2^{16}$， $0 \leqslant r \leqslant 2^{16}$。

易见：$0 \leqslant q + r < 2^{16} + 2^{16} = 2^{17}$，且 $ab = q \cdot 2^{16} + (q+r)$，所以：

$$\left[{ab}/{2^{16}} \right] = \begin{cases} q, & q+r<2^{16} \\ q+1, & q+2 \geqslant 2^{16} \end{cases} \tag{4-1}$$

$$\left(ab \mod 2^{16}\right) = \begin{cases} q+r, & q+r<2^{16} \\ q+r-2^{16}, & q+2 \geqslant 2^{16} \end{cases} \tag{4-2}$$

在式（4-2）中求 r，得 $r = \begin{cases} \left(ab \mod 2^{16}\right)-q, & q+r<2^{16} \\ \left(ab \mod 2^{16}\right)+2^{16}-q, & q+r \geqslant 2^{16} \end{cases}$ （4-3）

在式（4-1）中求出 q，代入（4-3），得到：

$$r = \begin{cases} \left(ab \mod 2^{16}\right)-\left[{ab}/{2^{16}} \right], & q+r<2^{16} \\ \left(ab \mod 2^{16}\right)-\left[{ab}/{2^{16}} \right]+2^{16}+1, & q+r \geqslant 2^{16} \end{cases}$$ 。

而 $q+r<2^{16}$ 当且仅当 $\left(ab \mod 2^{16}\right) \geqslant \left[{ab}/{2^{16}} \right]$ 时成立，所以定理成立。

ab 是两个字的乘积，结果是一个双字，而 $\left[{ab}/{2^{16}} \right]$ 是 ab 的高位字，即高 16 位，$\left(ab \mod 2^{16}\right)$ 是 ab 的低位字，即低 16 位。可见，模运算转换成了加、减法运算。

至于乘法运算 ab，最多需要 16 次移位加法运算。

IDEA 的一个突出优点是实现速度快，这是由它的两种基本运算决定的。

4.2.5 IDEA 实例

（1）设明文信息为 Message=american，二进制编码后为：

MessageBits=0110000101101101011001010111001001101001011000110110000101101110， 16 个比特一组分组后的十进制表示为：

MeassgeNumberBlock={24941, 25970, 26979, 24942}。

（2）设密钥为：Key=ideaencryptosyst，二进制编码后为：

KeyBits=01101001011001000110010101100001011001010110111001100011011110010
0111100101110000011101000110111101110011011110010111001101110100。

由密钥生成器生成的 52 个子密钥为：

0110100101100100	0110010101100001	0110010101101110	0110001101110010	0111100101110000	0111010001101111
0111001101111001	0111001101110100	1100001011001010	1101110011000110	1110010011110010	1110000011101000
1101111011100110	1111001011100110	1110100011010010	1100100011001010	1000110111001001	1110010111000001
1101000110111101	1100110111100101	1100110111010001	1010010110010001	1001010110000101	1001010110111001
1000001110100011	0111101110011011	1100101110011011	1010001101001011	0010001100101011	0000101100101011
0111001100011011	1001001111001011	0011011110010111	0011011101000110	1001011001000110	0101011000010110
0101011011100110	0011011100100111	1001011100000111	0100011011110111	1000110100101100	1000110010101100
0010110010101101	1100110001101110	0100111100101110	0000111010001101	1110111001101111	0010111001101110
0101100001011001	0101101110011000	1101110010011110	0101110000011101		

写成十进制形式为：

$$
\mathrm{SubKey}=\begin{pmatrix}
26980 & 25953 & 25966 & 25458 & 31088 & 29807 \\
29561 & 29556 & 49866 & 56518 & 58610 & 57576 \\
57062 & 62182 & 59602 & 51402 & 36297 & 58817 \\
53693 & 52709 & 52689 & 42385 & 38277 & 38329 \\
33699 & 31643 & 52123 & 41803 & 9003 & 2859 \\
29467 & 37835 & 14231 & 14150 & 38470 & 22038 \\
22246 & 14119 & 38663 & 18167 & 36140 & 36012 \\
11437 & 52334 & 20270 & 3725 & 61039 & 11886 \\
22617 & 23448 & 56478 & 23581 & &
\end{pmatrix}
$$

52 个解密子密钥为：

$$
\begin{pmatrix}
1011101111110111 & 1010010001101000 & 0010001101100010 & 0010001000010100 & 1110111001101111 & 0010111001101110 \\
0101011001111000 & 1011000011010010 & 0011001110010010 & 0111000110001001 & 1000110100101100 & 1000110010101100 \\
0011011111011001 & 0110100011111001 & 1100100011011001 & 1010101100101100 & 1001011001000110 & 0101011000010110 \\
0001111110101101 & 1100100001101001 & 0110110000110101 & 1010101010001110 & 0010001100101011 & 0000101100101011 \\
0100100111011111 & 0011010001100101 & 1000010001100101 & 0011011110001010 & 1001010110000101 & 1001010110111001 \\
1010101011000111 & 0011001000101111 & 0011001000011011 & 1110111110111101 & 1000110111001001 & 1110010111000001 \\
1001111101001000 & 0001011100101110 & 0000110100011010 & 0101000101110111 & 1110010011110010 & 1110000011101000 \\
1111110100001101 & 0011110100110110 & 1000110010001100 & 1100111111111010 & 0111100101110000 & 0111010001101111 \\
1101010000001011 & 1001101010011111 & 1001101010010010 & 0101000110000110 & &
\end{pmatrix}
$$

写成十进制形式为：

$$
\mathrm{InvSubKey}=\begin{pmatrix}
48119 & 42088 & 9058 & 8724 & 61039 & 11886 \\
22136 & 45266 & 13202 & 29065 & 36140 & 36012 \\
14297 & 26873 & 51417 & 43820 & 38470 & 22038 \\
8109 & 51305 & 27701 & 43662 & 9003 & 2859 \\
18911 & 13413 & 33893 & 14218 & 38277 & 38329 \\
43719 & 12847 & 12827 & 61373 & 36297 & 58817 \\
40776 & 5934 & 3354 & 20855 & 58610 & 57576 \\
64781 & 15670 & 35980 & 53242 & 31088 & 29807 \\
54283 & 39583 & 39570 & 20870 & &
\end{pmatrix}
$$

（3）加密：

对待加密信息的十进制分组：

x0={24941, 25970, 26979, 24942}，

加密的 8 轮及输出轮的输出为：

x1={20261, 6797, 18408, 18975}，

x2={39195, 28770, 44134, 39528}，

x3={7453, 64209, 10749, 54851}，

x4={29498, 38779, 36509, 17994}，

x5={63594, 29437, 4367, 35520}，

x6={36656, 42731, 40790, 35605}，

x7={12360, 37523, 14774, 12050}，

x8={2104, 30944, 7058, 41645}，

x9={6306, 30506, 21886, 24337}，

所以密文为：x9={6306, 30506, 21886, 24337}。

（4）解密：

对于密文：

Inv_x0={6306, 30506, 21886, 24337}，

解密的 8 轮及输出为：

Inv_x1={63548, 35044, 24321, 58942}，

Inv_x2={38022, 13917, 56850, 51382}，

Inv_x3={24957, 18598, 1736, 4747}，

Inv_x4={53423, 23096, 4886, 35033}，

Inv_x5={5007, 63438, 51382, 97}，

Inv_x6={29428, 38200, 25416, 40182}，

Inv_x7={58315, 2738, 36353, 47119}，

Inv_x8={39801, 52945, 51923, 50980}，

Inv_x9={24941, 25970, 26979, 24942}，

所以得到解密明文：Inv_x9={24941, 25970, 26979, 24942}。

4.3 美国高级数据加密标准 AES

1997 年 9 月 12 日，为了替代即将退役的 DES，美国国家标准与技术研究所（NIST）在《联邦记事》上发出了征集 AES 算法的公告。1999 年 3 月 22 日，公布了 5 个候选算法：MARS，RC6，Rijndael，SERPENT，Twofish。2000 年 10 月 2 日，由比利时密码学家 Joan Daemen 和 Vincent Rijmen 提交的 Rijndael 算法被确定为 AES 算法，该算法的名字是由两位提出者的名字合成的。

4.3.1 算法

Rijndael 算法是一种分组密码体制，其明文分组长度、密钥长度可以是 128 比特、192 比特、256 比特中的任意一个。按照密钥长度，分别记为 AES-128，AES-192，AES-256。

Rijndael 算法采用 Square 结构，这是一种类似幻方的结构，主要有列变换，行移位变换。

Rijndael 算法以字节（8 比特）、（双）字（32 比特）为处理单位，将明文分成 N_b 个字，密钥分成 N_k 个字，每个字为四个字节。令 $N=\max\{N_b,N_k\}+6$，则算法共进行一个初始轮和 N–1 轮变换及末轮变换。

记初始轮变换为 T_0，末轮变换为 T_N，中间各轮为 $T_1,T_2,\cdots,T_{N-1}$，

则加密变换为：$AES=T_N\circ T_{N-1}\cdots T_1\circ T_0$；

若令：

A_i：轮密钥为 k_i 的密钥加变换，也记为 A_{k_i}

H_M：矩阵为 M 的列混合运算；

R_C：移位向量为 $C=(c_0,c_1,c_2,c_3)$ 的行右移位操作；

$S_{u,v}$：参数为 u，v 的 S-盒变换。

则：

$T_0 = A_0$；

$T_i = A_i H_M R_C S_{u,v}$，$i = 1,2,\cdots,N-1$；

$T_N = A_N R_C S_{u,v}$；

4.3.2 基本运算

1．字节运算——有限域 GF(2⁸)上的运算

字节运算是 AES 的基本运算，一个字节被看作 GF(2^8)的元素。有限域 GF(2^8)可定义为三元组：(F，+，•)，其中：

$F=\{(b_7b_6b_5b_4b_3b_2b_1b_0)|b_i=0,1;\ i=0,1,\ldots,7\}$。

（1）F 中的加法"+"定义为：

$(a_7a_6a_5a_4a_3a_2a_1a_0)+(b_7b_6b_5b_4b_3b_2b_1b_0)=(c_7c_6c_5c_4c_3c_2c_1c_0)$，

$c_i = a_i \oplus b_i$，$i = 0,1,\cdots,7$，而 $\oplus$ 是比特异或。

事实上，加法就是字节的按位异或运算，例如：

$$\begin{array}{r} 1\ 1\ 0\ 0\ 0\ 1\ 1\ 0 \\ +\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1 \\ \hline 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1 \end{array}$$

所以：$11000110+10101101=01101011$。

（2）F 中的乘法"•"定义为：

$(a_7a_6a_5a_4a_3a_2a_1a_0)(b_7b_6b_5b_4b_3b_2b_1b_0)=(c_7c_6c_5c_4c_3c_2c_1c_0)$，其中：

$a(x)=a_7x^7+a_6x^6+a_5x^5+a_4x^4+a_3x^3+a_2x^2+a_1x+a_0$，

$b(x)=b_7x^7+b_6x^6+b_5x^5+b_4x^4+b_3x^3+b_2x^2+b_1x+b_0$，

$c(x)=c_7x^7+c_6x^6+c_5x^5+c_4x^4+c_3x^3+c_2x^2+c_1x+c_0$，

$m(x)=x^8+x^4+x^3+x+1$。

则：$c(x)=a(x)b(x) \mod m(x)$，

其中 $a(x)b(x)$ 是普通多项式乘法，但系数运算看作比特的乘法和异或运算，即看作最小域 $F_2=\{0,1\}$ 上的运算。可见乘法运算不如加法运算那么简单，这也是 AES 运算费时的地方。

例如：

$$\begin{aligned} &\left(x^6+x^5+x^4+x+1\right)\left(x^7+x^4+x^2+1\right) \\ &=x^{13}+x^{12}+x^{11}+x^{10}+x^9+3x^8+2x^7+2x^6+2x^7+2x^5+2x^4+x^3+x^2+x+1 \\ &=x^{13}+x^{12}+x^{11}+x^{10}+x^9+x^8+x^3+x^2+x+1 \\ &=\left(x^5+x^4+x^3+x^2+1\right)\left(x^8+x^4+x^3+x+1\right)+x^6+x^5+x^4 \end{aligned}$$

所以，$(01110011)(10010101)=(01110000)$。

其实，$a(x)b(x)$ 可以理解为二进制乘法，即表示成移位和异或运算：

$$
\begin{array}{ccccccccccccccc}
 & & & & & & & & 1&1&1&0&0&1&1\\
\times & & & & & & & 1&0&0&1&0&1&0&1\\
\hline
 & & & & & & & & 1&1&1&0&0&1&1\\
 & & & & & & & 1&1&1&0&0&1&1& \\
 & & & & & & 1&1&1&0&0&1&1& & \\
\oplus & & & & & 1&1&1&0&0&1&1& & & \\
\hline
 & 1&1&1&1&1&1&0&0&0&0&1&1&1&1
\end{array}
$$

模运算 $d(x) \bmod m(x)$，由于 F_2 上的加法与减法相同，所以模运算也成为移位和异或运算，但从高位开始：

$$
\begin{array}{ccccccccccccccc}
 & 1&1&1&1&1&1&0&0&0&0&1&1&1&1\\
\oplus & 1&0&0&0&1&1&0&1&1& & & & & \\
\hline
 & &1&1&1&0&0&0&1&1&0& & & & \\
\oplus & &1&0&0&0&1&1&0&1&1& & & & \\
\hline
 & & &1&1&0&1&1&1&0&1&1& & & \\
\oplus & & &1&0&0&0&1&1&0&1&1& & & \\
\hline
 & & & &1&0&1&0&0&0&0&0&1& & \\
\oplus & & & &1&0&0&0&1&1&0&1&1& & \\
\hline
 & & & & & &1&0&1&1&0&1&0&1&1\\
\oplus & & & & & &1&0&0&0&1&1&0&1&1\\
\hline
 & & & & & & & &1&1&1&0&0&0&0
\end{array}
$$

2．字运算——系数在有限域 GF(2⁸)上的多项式的运算

全体（双）字集合及运算构成一个环：$(R, +, \cdot)$，$R=\left\{(a_3a_2a_1a_0)\middle|\ a_i \in GF\left(2^8\right)\right\}$。

（1）$(a_3a_2a_1a_0)+(b_3b_2b_1b_0)=(c_3c_2c_1c_0)$，$c_i=a_i+b_i$，$i=0,1,2,3$。$c_i=a_i+b_i$ 是 $GF(2^8)$ 中的加法运算。

（2）$(a_3a_2a_1a_0)(b_3b_2b_1b_0)=(c_3c_2c_1c_0)$，其中：

$$
\begin{aligned}
&c_3x^3+c_2x^2+c_1x+c_0\\
&=\left(a_3x^3+a_2x^2+a_1x+a_0\right)\left(b_3x^3+b_2x^2+b_1x+b_0\right) \bmod x^4+1,
\end{aligned}
$$

其中多项式乘法视为普通多项式乘法，但系数的运算是 $GF(2^8)$ 中的运算。

易见：

$$
\begin{aligned}
c_0 &= a_0b_0+a_1b_3+a_2b_2+a_3b_1\\
c_1 &= a_1b_0+a_2b_3+a_3b_2+a_0b_1\\
c_2 &= a_2b_0+a_3b_3+a_0b_2+a_1b_1\\
c_3 &= a_3b_0+a_0b_3+a_1b_2+a_2b_1
\end{aligned}
$$

其中的加法和乘法是 $GF(2^8)$ 中的运算。

用矩阵表示：

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix}。$$

注意：因为模多项式 x^4+1 不是既约多项式，所以 R 中有些非零元素没有逆元，因此（R，+，•）不能成为域，只能是一个环。

4.3.3 基本部件

1．密钥加变换——AddRoundKey

AES 每一轮变换的结果称为态，每个态是一个 $4\times N_b$ 阶字节的矩阵

$$a=(a_{ij})_{4\times N_b}=\begin{pmatrix} a_{11},a_{12},\cdots,a_{1N_b} \\ a_{21},a_{22},\cdots,a_{2N_b} \\ a_{31},a_{32},\cdots,a_{3N_b} \\ a_{41},a_{42},\cdots,a_{4N_b} \end{pmatrix},$$

a_{ij} 为 $GF(2^8)$ 中元素，初始态就是明文。

每一轮的轮密钥也是一个同样阶的矩阵：

$$k_i=(k_{st})_{4\times N_b}=\begin{pmatrix} k_{11},k_{12},\cdots,k_{1N_b} \\ k_{21},k_{22},\cdots,k_{2N_b} \\ k_{31},k_{32},\cdots,k_{3N_b} \\ k_{41},k_{42},\cdots,k_{4N_b} \end{pmatrix},$$

$k_i \quad (i=0,1,\cdots,N_r)$ 由原始密钥 k 通过密钥扩展算法产生。

则密钥加变换定义为：

$$A_i(a)=\left(a_{ij}\right)+k_i=\begin{pmatrix} a_{11}+k_{11},a_{12}+k_{12},\cdots,a_{1N_b}+k_{1N_b} \\ a_{21}+k_{21},a_{22}+k_{22},\cdots,a_{2N_b}+k_{2N_b} \\ a_{31}+k_{31},a_{32}+k_{32},\cdots,a_{3N_b}+k_{3N_b} \\ a_{41}+k_{41},a_{42}+k_{42},\cdots,a_{4N_b}+k_{4N_b} \end{pmatrix}，$$ 其中 $a_{ij}+k_{ij}$ 是 $GF\left(2^8\right)$ 中的运算。

2．S-盒变换——ByteSub 变换

ByteSub 变换也称 S-盒变换，它由两个变换复合而成。

（1）倒数变换：t: $GF\left(2^8\right)\to GF\left(2^8\right)$，$t$: $a\mapsto t(a)$，

对 $\forall a\in GF\left(2^8\right)$， $t(a)=\begin{cases} 0, & a=0 \\ a^{-1}, & a\neq 0 \end{cases}$

（2）仿射变换：$L_{u,v}$: $GF\left(2^8\right)\to GF\left(2^8\right)$，$a:\mapsto L_{u,v}(a)$

对 $\forall a\in GF\left(2^8\right)$ ， $L_{u,v}(a)=b=(b_7b_6\cdots b_1b_0)$，

用多项式 $b(x)=b_0+b_1x+\cdots+b_7x^7$ 表示 $(b_7b_6\cdots b_1b_0)$，

用多项式 $a(x)=a_0+a_1x+\cdots+a_7x^7$ 表示 $(a_7a_6\cdots a_1a_0)$，则：

$b(x)=u(x)a(x)+v(x)\quad \bmod \quad x^8+1$

$u(x)=x^7+x^6+x^5+x^4+1$，$v(x)=x^7+x^6+x^2+x$。

写成系数形式为：
$$\begin{pmatrix} b_0\\ b_1\\ b_2\\ b_3\\ b_4\\ b_5\\ b_6\\ b_7 \end{pmatrix}=\begin{pmatrix} 1&1&1&1&1&0&0&0\\ 0&1&1&1&1&1&0&0\\ 0&0&1&1&1&1&1&0\\ 0&0&0&1&1&1&1&1\\ 1&0&0&0&1&1&1&1\\ 1&1&0&0&0&1&1&1\\ 1&1&1&0&0&0&1&1\\ 1&1&1&1&0&0&0&1 \end{pmatrix}\begin{pmatrix} a_0\\ a_1\\ a_2\\ a_3\\ a_4\\ a_5\\ a_6\\ a_7 \end{pmatrix}+\begin{pmatrix} 0\\ 1\\ 1\\ 0\\ 0\\ 0\\ 1\\ 1 \end{pmatrix}$$

记 ByteSub 变换为 $S_{u,v}$，则：$S_{u,v}=L_{u,v}\circ t$，

$$S_{u,v}(a(x))=\left(u(x)\left(a_{ij}^{-1}(x)\ \bmod\ m(x)\right)+v(x)\ \bmod\ x^8+1\right)_{4\times N_b}$$。

由此可见，虽然这两个变换都是对 $GF\left(2^8\right)$ 中的元素进行的，但是却使用了两种不同的数学结构，根本上，t 是在有限域 $GF\left(2^8\right)=F_2[x]/(m(x))$ 上进行的，但 $L_{u,v}$ 却是在环"$GF(2^8)$" $=F_2[x]/(x^8+1)$ 上进行的。虽然 t，$L_{u,v}$ 都非常简单，但它们的复合却非常复杂，这种集合相同，数学结构不同的运算的复合是 AES 的 S-盒变换非线性的保证，也是现代密码学采用的常用手段。如 IDEA 的模加、模乘运算。

$S_{u,v}$ 可用 $GF(2^8)$中的多项式表示，这个多项式可以通过某些性质计算得到，也可以直接用插值方法求得：

设 $P_{255}(x)=q_0+q_1x+q_2x^2+\cdots+q_{255}x^{255}$，令：

$P_{255}(a)=S_{u,v}(a)$，$a\in GF(2^8)$，得到 255 个线性方程组，解这个方程组，可以解得：

$P_{255}(x)=99+143x^{127}+181x^{191}+x^{223}+244x^{239}+37x^{247}+249x^{251}+9x^{253}+5x^{254}$。

$S_{u,v}$ 给出的 $GF(2^8)$到 $GF(2^8)$的变换表见表 4-4。

表 4-4 S-盒变换表的十六进制表示

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf

（续）

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

$S_{u,v}$的逆变换$S^{-1}_{u,v}$给出的$GF(2^8)$到$GF(2^8)$的变换表见表 4-5。

表 4-5　逆 S-盒变换表的十六进制表示

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

3．列混合变换——MixColumn 变换

列混合变换作用在中间态的列上，如果态的一个列记为系数在 GF(2^8)上的多项式：$t(x)=t_3x^3+t_2x^2+t_1x+t_0$，则列混合变换可表示为：

H_M：$t(x)\mapsto d(x)$，其中$d(x)=a(x)t(x) \mod x^4+1$。

$a(x)=\{03\}x^3+\{01\}x^2+\{01\}x+\{02\}$，而其中系数的乘法和加法都是 GF($2^8$)中的运算。

如果记 $d(x)=d_3x^3+d_2x^2+d_1x+d_0$，则有：

$$\begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{pmatrix}=\begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix}\begin{pmatrix} t_0 \\ t_1 \\ t_2 \\ t_3 \end{pmatrix}$$ 。记 $M=\begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix}$，$T$ 为中间态，

则列混合变换为：$H_M(T)=MT$ 。

4．行移位运算——ShiftRows 变换

行移位变换作用在中间态的行上，第 0 行不动，第 1 行循环左移 1 个字节，第 2 行循环左移 2 个字节，第 3 行循环左移 3 个字节，即行移位向量 $C=(0,1,2,3)$，

R_C：$a\mapsto R_C(a)$ 为：

$$\begin{pmatrix} a_{01} & a_{02} & \cdots & a_{0N_b-1} & t_{0N_b} \\ a_{11} & a_{12} & \cdots & a_{1N_b-1} & a_{1N_b} \\ a_{21} & a_{22} & \cdots & a_{2N_b-1} & a_{2N_b} \\ a_{31} & a_{32} & \cdots & a_{3N_b-1} & a_{3N_b} \end{pmatrix}\mapsto\begin{pmatrix} a_{01} & a_{02} & \cdots & a_{0N_b-1} & a_{0N_b} \\ a_{12} & a_{13} & \cdots & a_{1N_b} & a_{11} \\ a_{23} & a_{24} & \cdots & a_{21} & a_{22} \\ a_{34} & a_{35} & \cdots & a_{32} & a_{33} \end{pmatrix}$$

4.3.4 AES 解密原理

1．基本运算的逆

（1）A_i 并行地作用于态阵的列的每个字节上，所以，$A_i^{-1}=A_i$；

（2）H_M 并行地作用于态阵的列上，且$(H_M)^{-1}=M^{-1}T$，所以，$H_M^{-1}=H_{M^{-1}}$，M^{-1} 表示 GF(2^2)中矩阵 M 的逆阵；

（3）S-变换 $S_{u,v}$ 并行地作用于态阵的字节上，且 $S_{u,v}=L_{u,v}\circ t$ ，$t=t^{-1}$，$L_{u,v}=L_{u^{-1},-v}$，所以，$S_{u,v}^{-1}=S_{u^{-1},-v}$，其中 u^{-1}，$-v$ 表示模 x^8+1 逆元和负元，$-C=(-c_0,-c_1,-c_2,-c_3)$；

（4）显然：$R_C^{-1}=R_{C^{-1}}$ 。

2．基本运算的交换律

（1）S-盒变换是字节并行运算，右移运算不改变字节的值，所以，它们的运算顺数是可以交换的，于是得：$R_CS_{u,v}=S_{u,v}R_C$，那么自然有：$R_C^{-1}S_{u,v}^{-1}=S_{u,v}^{-1}R_C^{-1}$ 。

（2）$A_i(H_M(T))=A_i(M(T))$

$$=MT+k_i=MT+M(M^{-1}k_i)=H_M(A_{H_M^{-1}(k_i)}(T)),$$

所以：$A_i\circ H_M=H_M\circ A_{H_M^{-1}(k_i)}$，从而有：

$$H_M^{-1}A_i^{-1}=(A_iH_M)^{-1}=\left(H_MA_{H_M^{-1}(k_i)}\right)^{-1}=A_{H_M^{-1}(k_i)}^{-1}H_M^{-1}=A_{H_{M^{-1}}(k_i)}H_{M^{-1}} 。$$

3．解密原理

$$AES^{-1}=T_0^{-1}\circ T_1^{-1}\circ T_2^{-1}\circ\cdots\circ T_{N-2}^{-1}\circ T_{N-1}^{-1}\circ T_N^{-1}$$
$$=A_0^{-1}\circ S_{u,v}^{-1}R_C^{-1}H_M^{-1}A_1^{-1}\circ S_{u,v}^{-1}R_C^{-1}H_M^{-1}A_2^{-1}\circ\cdots\circ S_{u,v}^{-1}R_C^{-1}H_M^{-1}A_{N-1}^{-1}\circ S_{u,v}^{-1}R_C^{-1}A_1^{-1}$$

$$=A_0R_{-C}S_{u^{-1},-v}\circ A_{H_{M^{-1}}(k_1)}H_{M^{-1}}R_{-C}S_{u^{-1},-v}\circ\cdots\circ A_{H_{M^{-1}}(k_{N-2})}H_{M^{-1}}R_{-C}S_{u^{-1},-v}$$

$$\circ A_{H_{M^{-1}}(k_{N-1})}H_{M^{-1}}R_{-C}S_{u^{-1},-v}\circ A_N$$ 。

可见，解密算法与加密算法结构完全相同，所不同的是在解密时：

（1）轮密钥的使用顺序与加密过程相反，且除第一个和最后一个轮密钥外，其余轮密钥需要进行逆混合列变换。即：如果加密轮密钥为：$k_0,k_1,\cdots,k_{N-1},k_N$，那么，解密轮密钥为：$k_N,H_{M^{-1}}(k_{N-1}),\cdots,H_{M^{-1}}(k_1),k_0$；

（2）相应的 S-盒中仿射变换为原仿射变换的逆变换；

（3）右移位向量的分量为原右移位向量分量模 N_b 的加法逆 $-C=(-c_0,-c_1,-c_2,-c_3)$；

（4）混合变换中的变换矩阵为原矩阵的逆矩阵：$M^{-1}=\begin{pmatrix}0E & 0B & 0D & 09\\ 09 & 0E & 0B & 0D\\ 0D & 09 & 0E & 0B\\ 0B & 0D & 09 & 0E\end{pmatrix}$。

4.3.5 密钥生成器

设密钥块为 $K_1K_2\cdots K_{N_k}$，K_j 是一个字，$j=1,2,\cdots,N_k$，即 4 个字节。

通过生成器产生 N_r+1 轮轮密钥，每个轮密钥由 N_b 个字组成,共有 $N_b(N_r+1)$ 个字 W[i]，i=0,1,2,…,N_b*(N_r+1)-1。

N_k≤6 时，

```
KeyExpansion(K[4*Nb],W[Nb*(Nr+1)])
{for(j=0; j<Nk; j++)W[j]=(K[4*j], K[4*j+1], K[4*j+2], K[4*j+3]);
 for(j=Nk; j<Nb(Nr+1); j++)
    {temp=W[j-1];
     if(j%Nk==0)temp=SubByte(RotByte(temp))^Rcon[j/Nk];
     W[j]=W[j-Nk]^temp;
     }
}
```

算法如图 4-28 所示。

N_k>6 时，

```
KeyExpansion(K[4*Nb],W[Nb*(Nr+1)])
{for(j=0; j<Nk; j++)W[j]=(K[4*j], K[4*j+1], K[4*j+2], K[4*j+3]);
 for(j= Nk; j<Nb(Nr+1); j++)
    {temp=W[j-1];
     if(j%Nk==0)temp=SubByte(RotByte(temp))^Rcon[j/Nk];
else if(j%Nk==4)
    temp=SubByte(temp);
W[j]=W[j-Nk]^temp;
}
}
```

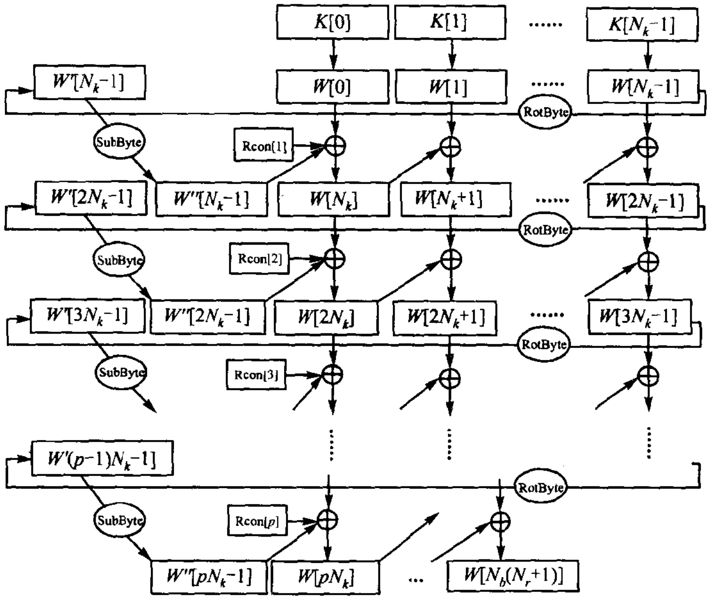


图 4-28　$N_k \leqslant 6$ 时 AES 密钥生成器

算法如图 4-29 所示。

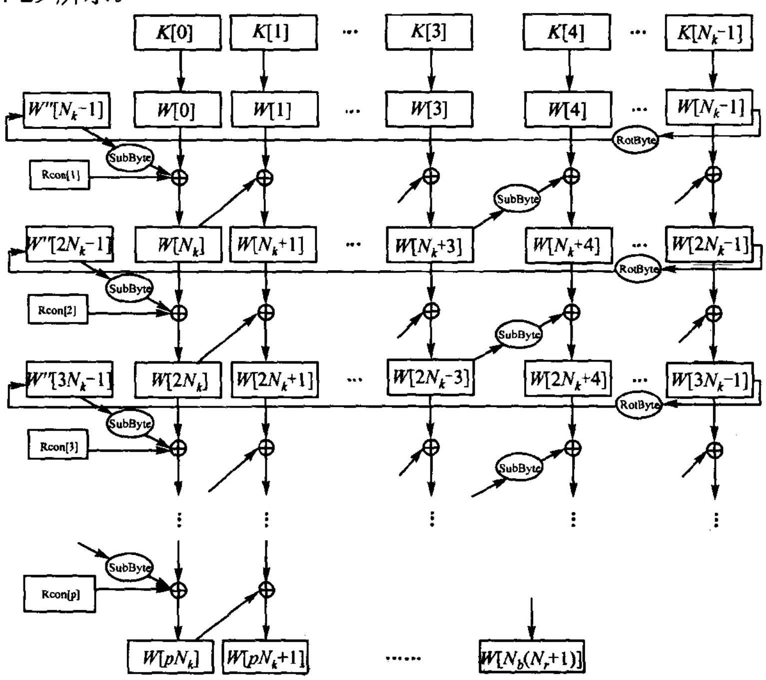


图 4-29　$N_k > 6$ 时 AES 密钥生成器

其中 $\text{Rcon}[i]=\left\{x^{i-1},0,0,0\right\}$ 是轮常数，其每个分量都视为 GF(2^8)中的元。

SubByte 是 S-Box 在 4 字节上的运用。RotByte 是字节右移 1 位操作，即：对 $\forall a,b,c,d \in$ GF(2^8)，RotByte$(a,b,c,d)=(b,c,d,a)$。

得到的子密钥字序列按顺序每轮取 N_b 个字，即轮密钥分别为：

首　轮：　$W[0],\ldots,W[N_b-1]$；

第一轮：　$W[N_b],\ldots,W[2N_b-1]$；

…

第 N_{r-1} 轮：$W[N_{r-1}N_b],\ldots,W[N_rN_b-1]$；

末　　轮：$W[N_rN_b],\ldots,W[(N_r+)N_b-1]$。

4.3.6 AES 实例

本例中约定用十六进制表示所有信息，考虑 AES-128。

输入信息 ＝32 43 f6 a8 88 5a 30 8d 31 31 98 a2 e0 37 07 34

加密密钥 ＝2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c

轮数 N_r=10。

计算过程见表 4-6。

表 4-6　AES-128 示例计算过程

轮序	轮输入				SubBytes 结果				ShiftRows 结果				MixColumns 结果					轮密钥				
输入	32	88	31	e0														2b	28	ab	09	
	43	5a	31	37													⊕	7e	ae	f7	cf	=
	f6	30	98	07														15	d2	15	4f	
	a8	8d	a2	34														16	a6	88	3c	
1	19	a0	9a	e9	d4	e0	b8	1e	d4	e0	b8	1e	04	e0	48	28		a0	88	23	2a	
	3d	f4	c6	f8	27	bf	b4	41	bf	b4	41	27	66	cb	f8	06	⊕	fa	54	a3	6c	=
	e3	e2	8d	48	11	98	5d	52	5d	52	11	98	81	19	d3	26		fe	2c	39	76	
	be	2b	2a	08	ae	f1	e5	30	30	ae	f1	e5	e5	9a	7a	4c		17	b1	39	05	
2	a4	68	6b	02	49	45	7f	77	49	45	7f	77	58	1b	db	1b		f2	7a	59	73	
	9c	9f	5b	6a	de	db	39	02	db	39	02	de	4d	4b	e7	6b	⊕	c2	96	35	59	=
	7f	35	ea	50	d2	96	87	53	87	53	d2	96	ca	5a	ca	b0		95	b9	80	f6	
	f2	2b	43	49	89	f1	1a	3b	3b	89	f1	1a	f1	ac	a8	e5		f2	43	7a	7f	
3	aa	61	82	68	ac	ef	13	45	ac	ef	13	45	75	20	53	bb		3d	47	1e	6d	
	8f	dd	d2	32	73	c1	b5	23	c1	b5	23	73	ec	0b	c0	25	⊕	80	16	23	7a	=
	5f	e3	4a	46	cf	11	d6	5a	d6	5a	cf	11	09	63	cf	d0		47	fe	7e	88	
	03	ef	d2	9a	7b	df	b5	b8	b8	7b	df	b5	93	33	7c	dc		7d	3e	44	3b	
4	48	67	4d	d6	52	85	e3	f6	52	85	e3	f6	0f	60	6f	5e		ef	a8	b6	db	
	6c	1d	e3	5f	50	a4	11	cf	a4	11	cf	50	d6	31	c0	b3	⊕	44	52	71	0b	=
	4e	9d	b1	58	2f	5e	c8	6a	c8	6a	2f	5e	da	38	10	13		a5	5b	25	ad	
	ee	0d	38	e7	28	d7	07	94	94	28	d7	07	a9	bf	6b	01		41	7f	3b	00	
5	e0	c8	d9	85	e1	e8	35	97	e1	e8	35	97	25	bd	b6	4c		d4	7c	ca	11	
	92	63	b1	b8	4f	fb	c8	6c	fb	c8	6c	4f	d1	11	3a	4c	⊕	d1	83	f2	f9	=
	7f	63	35	be	d2	fb	96	ae	96	ae	d2	fb	a9	d1	33	c0		c6	9d	b8	15	
	e8	c0	50	01	9b	ba	53	7c	7c	9b	ba	53	ad	68	8e	b0		f8	87	bc	bc	

（续）

6	f1	c1	7c	5d	a1	78	10	4c	a1	78	10	4c	4b	2c	33	37	⊕	6d	11	db	ca	=
	00	92	c8	b5	63	4f	e8	d5	4f	e8	d5	63	86	4a	9d	d2		88	0b	f9	00	
	6f	4c	8b	d5	a8	29	3d	03	3d	03	a8	29	8d	89	f4	18		a3	3e	86	93	
	55	ef	32	0c	fc	df	23	fe	fe	fc	df	23	6d	80	e8	d8		7a	fd	41	fd	
7	26	3d	e8	fd	f7	27	9b	54	f7	27	9b	54	14	46	27	34	⊕	4e	5f	84	4e	=
	0e	41	64	d2	ab	83	43	b5	83	43	b5	ab	15	16	46	2a		54	5f	a6	a6	
	2e	b7	72	8b	31	a9	40	3d	40	3d	31	a9	b5	15	56	d8		f7	c9	4f	dc	
	17	7d	a9	25	f0	ff	d3	3f	3f	f0	ff	d3	bf	ec	d7	43		0e	f3	b2	4f	
8	5a	19	a3	7a	be	d4	0a	da	be	d4	0a	da	00	b1	54	fa	⊕	ea	b5	31	7f	=
	41	49	e0	8c	83	3b	e1	64	3b	e1	64	83	51	c8	76	1b		d2	8d	2b	8d	
	42	dc	19	04	2c	86	d4	f2	d4	f2	2c	86	2f	89	6d	99		73	ba	f5	29	
	b1	1f	65	0c	c8	c0	4d	fe	fe	c8	c0	4d	d1	ff	cd	ea		21	d2	60	2f	
9	ea	04	65	85	87	f2	4d	97	87	f2	4d	97	47	40	a3	4c	⊕	ac	19	28	57	=
	83	45	5d	96	ec	6e	4c	90	6e	4c	90	ec	37	d4	70	9f		77	fa	d1	5c	
	5c	33	98	b0	4a	c3	46	e7	46	e7	4a	c3	94	e4	3a	42		66	dc	29	00	
	f0	2d	ad	c5	8c	d8	95	a6	a6	8c	d8	95	ed	a5	a6	bc		f3	21	41	6e	
10	eb	59	8b	1b	e9	cb	3d	af	e9	cb	3d	af					⊕	d0	c9	e1	b6	=
	40	2e	a1	c3	09	31	32	2e	31	32	2e	09						14	ee	3f	63	
	f2	38	13	42	89	07	7d	2c	7d	2c	89	07						f9	25	0c	0c	
	1e	84	e7	d2	72	5f	94	b5	b5	72	5f	94						a8	89	c8	a6	

输出	39	02	dc	19
	25	dc	11	6a
	84	09	85	0b
	1d	fb	97	32

4.4 欧洲密码标准

4.4.1 引言

新欧洲数字签名、数据完整性、数据加密计划 NESSIE (New European Schemes for Signature, Integrity, and Encryption) 是“欧洲共同体信息社会技术计划”（ISTP）中的一项工程，该工程的主要目标就是通过公开征集和进行公开的、透明的测试评价提出一套强壮的密码标准。这些标准包括分组密码，流密码，散列函数，消息认证码（MAC），数字签名和公钥加密。该工程将提供一种密码体制评估方法(包括安全性和性能评价) 和支持评估的一个软件工具箱。该工程的另一个目标就是广泛传播这些工程成果，保持欧洲工业界在密码学研究领域的领先地位。

NESSIE 于 2000 年 1 月启动，分为两个阶段，为期 3 年。第一阶段始于 2000 年 1 月，终于 2001 年 6 月，第二阶段始于 2000 年 7 月终于 2002 年 12 月。

NESSIE 第一次专题研讨会于 2000 年 11 月 13 日~14 日进行。递交会议的共有 35 个方案，会议对应征算法进行初步筛选。第二次工作会议于 2001 年 9 月 12 日~13 日进行，进一步对算法评估，筛选。第三次工作会议于 2002 年 11 月 6 日~7 日进行，确定最后的各类算法，2003 年 2 月将召开第四次工作会议。

经过第 1、2 轮筛选，确定的候选标准有：

（1）一般安全水平的分组密码算法：

IDEA，Khazad，MISTY1，SAFER++64

（2）标准、高级安全水平分组密码算法：

SAFER++128，Camellia，RC6，SHACAL

（3）消息认证 MAC 和散列函数算法：

Two-Track-MAC，UMAC，Whirlpool

（4）流密码：

SOBER-t16、SOBER-t32，SNOW，BMGL

（5）公钥密码算法：

ACE (ACE-KEM 的修订版)，EPOC-2 (修订版)，PSEC-2 (PSEC-KEM 的修订版)，ECIES，RSA-OAEP (需要修订)

（6）数字签名算法：

ECDSA，ESIGN (修订版)，RSA-PSS，SFLASH，QUARTZ (相对于具体应用)

（7）数据鉴别算法：

GPS

这些密码算法将成为近十年密码研究、应用的重要内容。下面仅就其中的分组密码算法 Khazad，RC6，Safer++进行介绍。

4.4.2 Khazad

Khazad 由巴西的 P.Barreto 和比利时的 V.Rijmen 联合设计。其明文分组为 64 比特，密钥长度为 128 比特，采用 SP 网络结构，算法共 8 轮。

1．Khazad 的基础空间

Khazad 的基础空间为有限域：

$$GF\left(2^8\right)=Z_2[x]\Big/\left(m(x)\right),\quad m(x)=x^8+x^4+x^3+x^2+1$$

及其上的向量空间：$GF\left(2^8\right)^8=\left\{(a_0,a_1,\cdots,a_7)\mid a_i\in GF\left(2^8\right),i=0,1,\cdots,7\right\}$。

2．S 盒变换

S 盒是 $GF\left(2^8\right)$ 上的一个置换，S：$GF\left(2^8\right)\to GF\left(2^8\right)$，如表 4-7 所示。

表 4-7 S 盒置换表

XY	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
00	A7	D3	E6	71	D0	AC	4D	79	3A	C9	91	FC	1E	47	54	BD
01	8C	A5	7A	FB	63	B8	DD	D4	E5	B3	C5	BE	A9	88	0C	A2
02	39	DF	29	DA	2B	A8	CB	4C	4B	22	AA	24	41	70	A6	F9
03	5A	E2	B0	36	7D	E4	33	FF	60	20	08	8B	5E	AB	7F	78
04	7C	2C	57	D2	DC	6D	7E	0D	53	94	C3	28	27	06	5F	AD
05	67	5C	55	48	0E	52	EA	42	5B	5D	30	58	51	59	3C	4E
06	38	8A	72	14	E7	C6	DE	50	8E	92	D1	77	93	45	9A	CE
07	2D	03	62	B6	B9	BF	96	6B	3F	07	12	AE	40	34	46	3E
08	DB	CF	EC	CC	C1	A1	C0	D6	1D	F4	61	3B	10	D8	68	A0
09	B1	0A	69	6C	49	FA	76	C4	9E	9B	6E	99	C2	B7	98	BC
10	8F	85	1F	B4	F8	11	2E	00	25	1C	2A	3D	05	4F	7B	B2
11	32	90	AF	19	A3	F7	73	9D	15	74	EE	CA	9F	0F	1B	75
12	86	84	9C	4A	97	1A	65	F6	ED	09	BB	26	83	EB	6F	81
13	04	6A	43	01	17	E1	87	F5	8D	E3	23	80	44	16	66	21
14	FE	D5	31	D9	35	18	02	64	F2	F1	56	CD	82	C8	BA	F0
15	EF	E9	E8	FD	89	D7	C7	B5	A4	2F	95	13	0B	F3	E0	37

可以验证 $S^2 = id$ 。

3．非线性层变换 γ

γ ： $GF\left(2^8\right)^8 \to GF\left(2^8\right)^8$ ，定义为： $\gamma(a_0, a_1, \cdots, a_7) = \left(S(a_0), S(a_1), \cdots, S(a_7)\right)$ ，由于 $S^2 = id$ ，所以也有 $\gamma^2 = id$ 。

4．线性混合层 θ

θ ： $GF\left(2^8\right)^8 \to GF\left(2^8\right)^8$ ，定义为：

θ ： $x \mapsto y = \theta(x)$ ，

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{pmatrix} = \begin{pmatrix} 01 & 03 & 04 & 05 & 06 & 08 & 0b & 07 \\ 03 & 01 & 05 & 04 & 08 & 06 & 07 & 0b \\ 04 & 05 & 01 & 03 & 0b & 07 & 06 & 08 \\ 05 & 04 & 03 & 01 & 07 & 0b & 08 & 06 \\ 06 & 08 & 0b & 07 & 01 & 03 & 04 & 05 \\ 08 & 06 & 07 & 0b & 03 & 01 & 05 & 04 \\ 0b & 07 & 06 & 08 & 04 & 05 & 01 & 03 \\ 07 & 0b & 08 & 06 & 05 & 04 & 03 & 01 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix},$$

5．密钥加变换 $\sigma[k]$

对于任意密钥向量 $k = (k_0, k_1, \cdots, k_7) \in GF\left(2^8\right)^8$ ，

$\sigma[k]$ ： $GF\left(2^8\right)^8 \to GF\left(2^8\right)^8$ ，定义为：

$\sigma[k]$ ： $(a_0, a_1, \cdots, a_7) \mapsto \sigma[k](a_0, a_1, \cdots, a_7) = (a_0 + k_0, a_1 + k_1, \cdots, a_7 + k_7)$ 。

6. 轮常数

第 r 轮的轮常数定义为：

$c^r=\left(S(8r+0),S(8r+1),\cdots,S(8r+7)\right)\in GF\left(2^8\right)^8$， $r=0,1,\cdots,7$。

7. 轮函数 $\rho[k]$

对于任意密钥向量 $k=(k_0,k_1,\cdots,k_7)\in GF\left(2^8\right)^8$，

$\rho[k]$： $GF\left(2^8\right)^8\rightarrow GF\left(2^8\right)^8$，定义为：

$\rho[k]\equiv\sigma[k]\circ\theta\circ\gamma$。

8. 子密钥生成过程

设主密钥 $k\in GF\left(2^8\right)^{16}$，其左、右 64 比特分别记为 k_L， k_R。

$k^{(-2)}=k_L$， $k^{(-1)}=k_R$， $k^{(i)}=\rho\left[c^i\right]\left(k^{(i-1)}\right)\oplus k^{(i-2)}$， $i=0,1,\cdots,8$。

9. 加密过程

Khazad 的加密过程首先将 64 比特明文块分成 8 个 8 比特的字节。

$$
\begin{aligned}
\text{Khazad}[\text{K}]&=\alpha_r\left[k^0,k^1,\cdots,k^r\right]\\
&=\sigma[k]\circ\gamma\circ\rho[k^{r-1}]\circ\cdots\circ\rho[k^1]\circ\sigma[k^0]\\
&=\sigma[k]\circ\gamma\circ\left(\sigma[k^{r-1}]\circ\theta\circ\gamma\right)\circ\cdots\circ\left(\sigma[k^1]\circ\theta\circ\gamma\right)\circ\sigma[k^0]，\quad r=7。
\end{aligned}
$$

如图 4-30 所示：

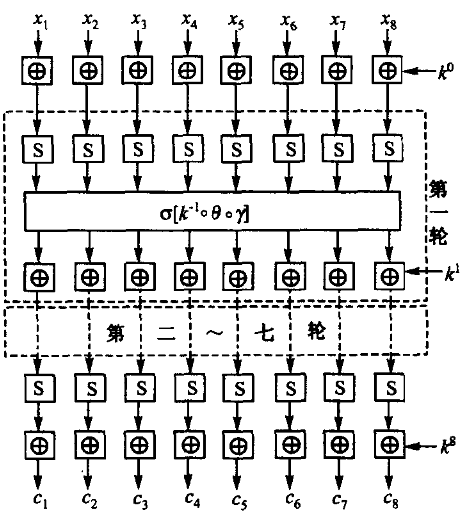


图 4-30　Khazad 加密过程

10. 解密过程

Khazad 的解密过程与加密过程完全一样，只是解密用的子密钥顺序与加密子密钥的顺序正好相反，即为： $k^8,k^7,\cdots,k^1,k^0$。

4.4.3 RC6

1994 年 RSA 公司的 R.L.Rivest 设计了分组迭代密码算法 RC5，但是，研究中发现了 RC5 的许多缺点，于是产生了改进版本 RC6，RC6 曾经是 AES 最终五个候选算法之一。

RC6 明文分组为 128 比特，密钥长度任意。设字长为 w 比特，加密轮数为 r，密钥的字节数（每个字节以 8 比特记）为 b，用 $RC6/w/r/b$ 来表示一个具体的 RC6 算法。

1．RC6 的 6 种基本字运算

逐比特异或：$\oplus$；

模 2^w 加法：+；

模 2^w 减法：−；

模 2^w 乘法：×；

字的循环左移：<<<；

字的循环右移：>>>；

2．RC6 加密过程

分割 128 比特明文消息为 4 个 32 比特的字 A,B,C,D；

设 $K[0]$，$K[1],\cdots$，$K[2r+3]$ 是由密钥生成器产生的 $2r+4$ 个子密钥；

加密算法：

（1）$B=B+K[0],D=D+K[1]$；

（2）对 $j=1,2,\cdots,r$ 做

$$t=\left(B\times(2B+1)\right)<<<\log_2 w,$$

$$u=\left(D\times(2D+1)\right)<<<\log_2 w,$$

$$A=\left((A\oplus t)<<<u\right)+K[2j],$$

$$C=\left((C\oplus u)<<<t\right)+K[2j+1],$$

$$(A,B,C,D)=(B,C,D,A),$$

$$A=A+K[2j+2],C=C+K[2j+3]。$$

（3）输出 (A,B,C,D)

主要轮变换如图 4-31 所示：

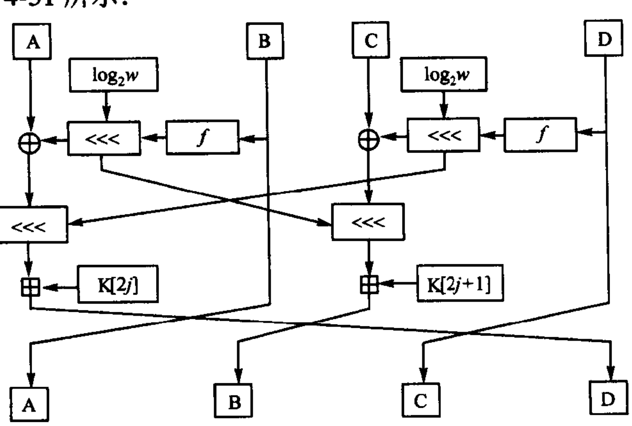


图 4-31 RC6 轮变换图

其中 $f(x)=x\times(2x+1) \mod 2^w$。

3．RC6 解密过程

分割 128 比特密文消息为 4 个 32 比特的字 A,B,C,D；

设 $K[0]$，$K[1],\cdots$，$K[2r+3]$是解密密钥（同加密密钥）；

解密算法：

（1）$C=C-K[2r+3], A=A-K[2r+2]$；

（2）对 $j=1,2,\cdots,r$ 做

$$(A,B,C,D)=(D,A,B,C)$$

$$u=(D\times(2D+1)) <<< \log_2 w,$$

$$t=(B\times(2B+1)) <<< \log_2 w,$$

$$C=((C-K[2(r-j)+3]) >>> t)\oplus u,$$

$$A=((A-K[2(r-j)+2]) >>> u)\oplus t,$$

$$D=D-K[1], B=B-K[0]。$$

（3）输出 (A,B,C,D)

4．密钥生成

设原始密钥 k 有 b 个字节（$0\leqslant b\leqslant 255$）。填充 k 使具有字长的整数倍（如：可以在右边添 0），填充后得到 c 个字，$k= L[0]\ L[1]\ \ldots\ L[c\text{-}1]$。

（1）设 P_w, Q_w 是两个确定的常数，如：$P_{32}=B7E15163, Q_{32}=9E3779B9$；

（2）令 $K[0]=P_w$；

（3）对 $j=1,2,\cdots,2r+3$ 做：$K[j]=K[j-1]+Q_w$；

（4）令 $A=B=i=j=0$，$v=3\times\max\{c,2r+4\}$；

（5）对 $s=1,2,\cdots,v$ 做：

$$A=K[i]=(K[i]+A+B) <<< 3,$$

$$B=L[j]=(L[j]+A+B) <<< (A+B),$$

$$i=i+1 \mod 2r+4,$$

$$j=j+1 \mod c;$$

（6）输出 $K[0],K[1],\cdots,K[2r+3]$。

4.4.4 Safer++加密算法

Safer++密码算法是 Cylink 公司提交的“新欧洲数字签名、完整性、加密标准”的候选算法。

该算法是 Safer 密码算法族：Safer K-64，Safer K-128，Safer SK-64，Safer SK-128，Safer SK-40，Safer +的改进版本。系列算法“Safer K”，“Safer SK”的分组大小都为 64 比特，密钥长度为 40、64、128 比特。Safer++的分组为 128 比特，密钥长度为 128、256 比特。Safer++曾是美国 AES 的五个候选算法之一。并被采用作为无线通信蓝牙协议的“挑战/响应”（challenge/response）实体认证方案。

“Safer K”，“Safer SK”系列是无产权的密码算法，由 ETH Zürich（Swiss Federal Institue

of Technology, Zürich）的 James L. Massey 教授在 Cylink 公司的请求下设计的。同样，Cylink 公司也声明放弃对 Safer++的所有权。

Safer 由 Massey 命名，是“Secure And Fast Encryption Routine”的首字母。第一个算法 Safer K-64 于 1993 年 11 月 9-11 日在英国剑桥召开的“Fast Software Encryption Workshop”专题大会上发表。系列的其他算法仅在密钥生成方案和加密轮数上有所不同。

初期的 Safer 算法族采用“2-点伪哈达玛变换（2-PHT）”的线性变换来实现“扩散”。Safer++在 Safer+的基础上作了重大的改进。Safer++比 Safer+更快，更简单，密码学意义更强壮。Safer++的改进由 Massey，Khachatrian，Kuregian 共同完成。Safer++新的主要特点是使用了 4-点伪哈达玛变换 4-PHT（Pseudo Hardamard Transform），取代了前面所有版本中使用的 2-PHT。

Safer++提供了三种水平：

高级水平具有：128 比特明文分组，256 比特密钥长度；

中级水平具有：128 比特明文分组，128 比特密钥长度；

标准水平具有：64 比特明文分组，128 比特密钥长度。

1．加、解密算法

和 DES 算法一样，Safer++也采用多轮迭代方法，记 Safer++的轮数为 r，密钥长度为 *Keylength*，则：*Keylength*=128 比特时，$r=7$，*Keylength*=256 比特时，$r=10$。

在每一加密轮中，使用了两个 16 字节的轮密钥。最后输出时使用了一个 16 字节的轮密钥，所以共需要 $2r+1$ 个轮密钥：（$K_1,K_2,\ldots,K_{2r},K_{2r+1}$），这些轮密钥由密钥生成器从用户密钥生成。

Safer++中的主要运算为：

⊕：$x \oplus y$ 表示逐比特异或运算，即逐比特模 2 加法运算；

⊞：$x \boxplus y= x+y \bmod 256$；

⊟：$x \boxminus y= x-y \bmod 256$；

exp：$x=128$ 时，$\exp(x)=0$，否则 $\exp(x)= 45^x \bmod 257$；

log：$x=0$ 时，$\log(x)=128$，否则 $\log(x)= \log_{45}(x)$。

则加密过程如图 4-32 所示：

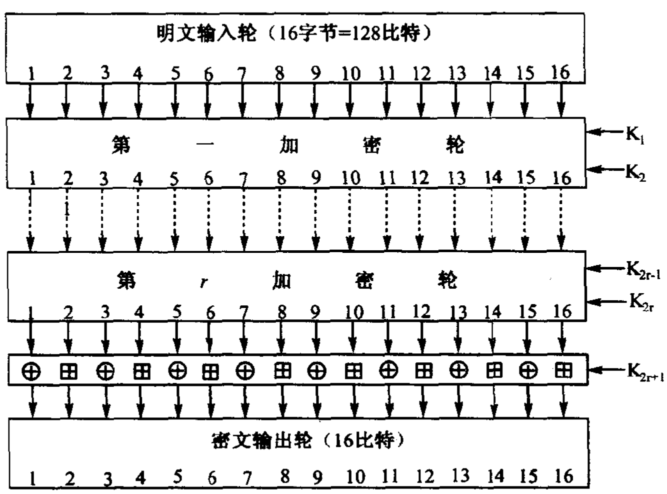


图 4-32　Safer++128 加密算法结构图

解密算法是将加密算法完全颠倒，解密过程如图 4-33 所示：

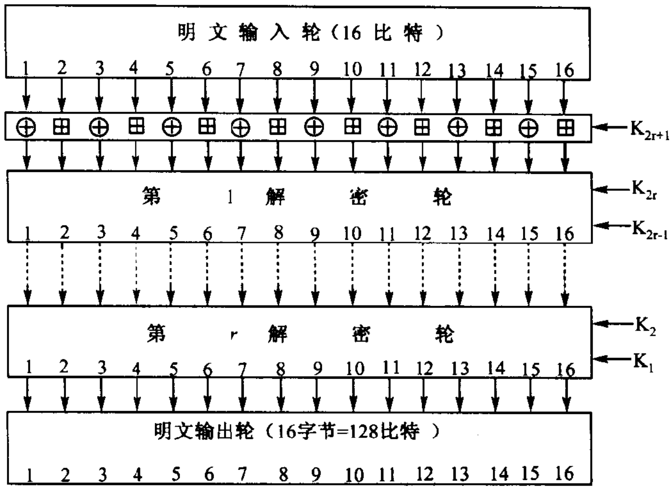


图 4-33 Safer++128 解密算法结构图

2. 轮加密变换

轮变换主要由钥控置换和可逆线性变换组成，设两个轮密钥为 K_{2i-1}，K_{2i}，则：

（1）钥控置换

1）密钥 K_{2i-1} 的加：

第 1，4，5，8，9，12，13，16 与子密钥相应字节作模 2 逐比特加法。

第 2，3，6，7，10，11，14，15 与子密钥相应字节作模 256 加法。

2）非线性层：

第 1，4，5，8，9，12，13，16 字节作运算 exp。

第 2，3，6，7，10，11，14，15 字节作运算 log。

3）密钥 K_{2i} 的加：

第 1，4，5，8，9，12，13，16 与子密钥相应字节作模 256 加法。

第 2，3，6，7，10，11，14，15 与子密钥相应字节作模 2 逐比特加法。

（2）可逆线性变换

1）预置乱置换：

对字节的序号作置换

$$P=\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16\\ 9 & 6 & 3 & 16 & 1 & 14 & 11 & 8 & 5 & 2 & 15 & 12 & 13 & 10 & 7 & 4\end{pmatrix}。$$

2）4-PHT 变换：

将第 1~4，5~8，9~12，13~16 字节分为一组，分别作为 4-PHT 的输入，设 (a,b,c,d) 是 4 字节的输入，(A,B,C,D) 是 4 字节的输出，则 4-PHT 变换如下：

$$(A,B,C,D)=(a,b,c,d)H_4,\quad H_4=\begin{pmatrix}2 & 1 & 1 & 1\\ 1 & 2 & 1 & 1\\ 1 & 1 & 2 & 1\\ 1 & 1 & 1 & 1\end{pmatrix}。$$

其中运算是模 256 的运算，即：

$A=2a+b+c+d \bmod 256$，$B=a+2b+c+d \bmod 256$，

$C=a+b+2c+d \mod 256$，$D=a+b+c+2d \mod 256$。

也可记为：

$A=a\boxplus a\boxplus b\boxplus c\boxplus d$，$B=a\boxplus b\boxplus b\boxplus c\boxplus d$，$C=a\boxplus b\boxplus c\boxplus c\boxplus d$，$D=a\boxplus b\boxplus c\boxplus d\boxplus d$。

于是，4-PHT 变换如图 4-34 所示。

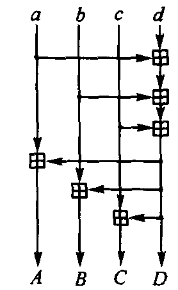


图 4-34　4-PHT 运算

3）中间置乱置换

对字节序号作置换：

$$P=\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16\\ 9 & 6 & 3 & 16 & 1 & 14 & 11 & 8 & 5 & 2 & 15 & 12 & 13 & 10 & 7 & 4\end{pmatrix}。$$

4）4 组 4-PHT 变换。

加密轮过程如图 4-35 所示。

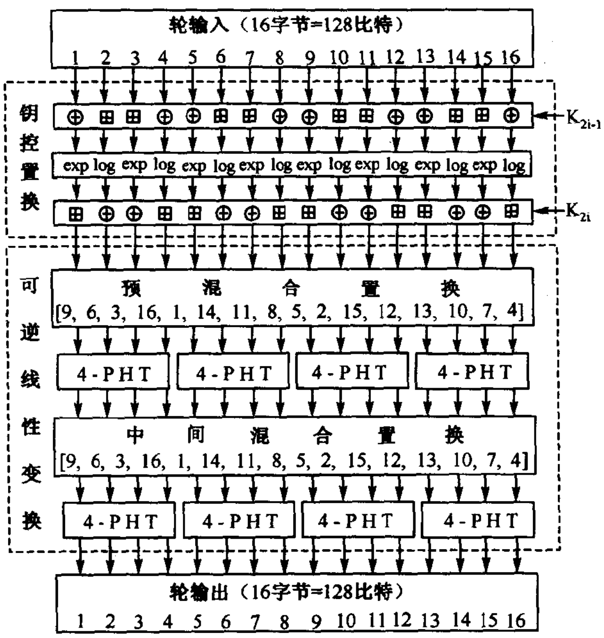


图 4-35　Safer++128 加密轮变换结构

3. 轮解密变换

设两个轮密钥为 $K_{2r-2i+2}$，K_{2i-1}，则：

（1）可逆非线性逆变换

1）4-IPHT 变换为：

将第 1-4，5-8，9-12，13-16 字节分为一组，分别作为 4-IPHT 的输入，

设 (A,B,C,D) 是 4 字节的输入，(a,b,c,d) 是 4 字节的输出，则：

$$(a,b,c,d)=(A,B,C,D)H_4^{-1}，\quad H_4^{-1}=\begin{pmatrix}1 & 0 & 0 & -1\\ 0 & 1 & 0 & -1\\ 0 & 0 & 1 & -1\\ -1 & -1 & -1 & 4\end{pmatrix}。$$

其中运算是模 256 的运算。即：

$A=a-d \mod 256$，$B=b-d \mod 256$，$C=c-d \mod 256$，$D=-a-b-c+4d \mod 256$，也可记为：

$A=a\boxminus d \bmod 256$，$B=b\boxminus d \bmod 256$，

$C=c\boxminus d \bmod 256$，$D=d\boxminus d\boxminus d\boxminus d\boxminus a\boxminus b\boxminus c \bmod 256$。

此过程如图 4-36 所示。

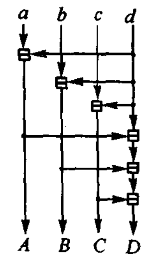


图 4-36 4-IPHT 运算

2）中间置乱逆置换：

对字节序号作置换

$$IVP=\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16\\ 5 & 10 & 3 & 16 & 9 & 2 & 15 & 8 & 1 & 14 & 7 & 12 & 13 & 6 & 11 & 4\end{pmatrix}。$$

3）4-IPHT 变换。

4）预置乱逆置换：即

对字节序号作置换

$$IVP=\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16\\ 5 & 10 & 3 & 16 & 9 & 2 & 15 & 8 & 1 & 14 & 7 & 12 & 13 & 6 & 11 & 4\end{pmatrix}。$$

（2）钥控逆变换

1）密钥 $K_{2r-2i+2}$ 减法

第 1，4，5，8，9，12，13，16 与子密钥相应字节作模 256 减法，

第 2，3，6，7，10，11，14，15 与子密钥相应字节作模 2 逐比特加法。

2）非线性层：

第 1，4，5，8，9，12，13，16 字节作运算 log。

第 2，3，6，7，10，11，14，15 字节作运算 exp。

3）密钥 K_{2i-1} 的加：

第 1，4，5，8，9，12，13，16 与子密钥相应字节作模 2 逐比特加法，

第 2，3，6，7，10，11，14，15 与子密钥相应字节作模 256 加法。

解密轮变换是加密轮变换的直接颠倒，如图 4-37 所示。

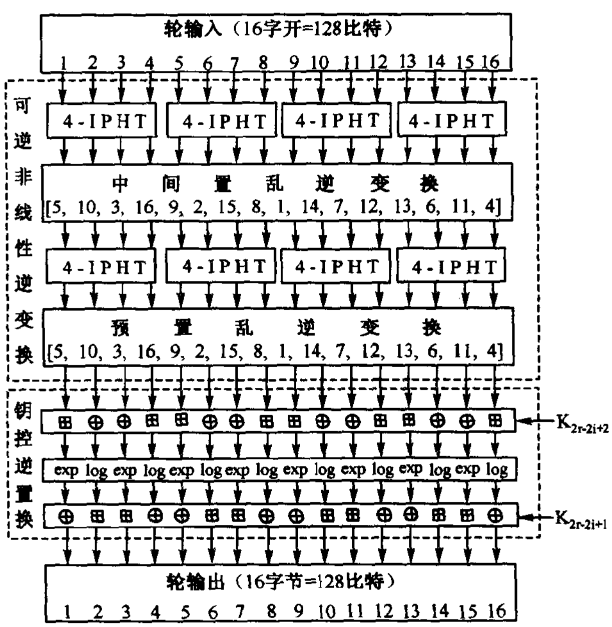


图 4-37 Safer++128 解密轮变换结构

4．密钥生成过程

r 轮加密过程中共需要 $2r+1$ 个子密钥。每个子密钥由 16 个字节共 128 比特组成。

如果初始密钥为 128 比特，则使用 15 个初始字节来生成子密钥；如果初始密钥为 256 比特，则使用 21 个字节。第一个字节仅仅为了便于表示，并不真正使用。

设 $B_i, i=1,2,3,\cdots,2r+1$ 表示 128 比特长度密钥的 $2r+1$ 个初始字。

令 $B_{i,j}$ 表示 B_i 的第 j 个字节。如果密钥长度为 128 比特，则：

$$B_{i,j}=45^{\left(45^{17i+j} \bmod 257\right)} \bmod 257\text{，}\quad i=2,3,\cdots,15\text{，}\quad j=1,2,\cdots,16\text{，}$$

如果等式右边为 256，则令 $B_{i,j}=0$。

对于 256 比特长度用户密钥，进一步定义 $B_{16},B_{17},\cdots,B_{21}$ 为：

$$B_{i,j}=45^{17i+j} \bmod 257\text{，}\quad i=16,\cdots,21\text{，}\quad j=1,2,\cdots,16\text{。}$$

如果等式右边为 256，则令 $B_{i,j}=0$。

计算得：

B_2= { 70, 151, 177, 186, 163, 183, 16, 10, 197, 55, 179, 201, 90, 40, 172, 100}
B_3= { 236, 171, 170, 198, 103, 149, 88, 13, 248, 154, 246, 110, 102, 220, 5, 61}
B_4= { 138, 195, 216, 137, 106, 233, 54, 73, 67, 191, 235, 212, 150, 155, 104, 160}
B_5= { 93, 87, 146, 31, 213, 113, 92, 187, 34, 193, 190, 123, 188, 153, 99, 148}
B_6= { 42, 97, 184, 52, 50, 25, 253, 251, 23, 64, 230, 81, 29, 65, 68, 143}
B_7= { 221, 4, 128, 222, 231, 49, 214, 127, 1, 162, 247, 57, 218, 111, 35, 202}
B_8= { 58, 208, 28, 209, 48, 62, 18, 161, 205, 15, 224, 168, 175, 130, 89, 44}
B_9= { 125, 173, 178, 239, 194, 135, 206, 117, 6, 19, 2, 144, 79, 46, 114, 51}
B_{10}={ 192, 141, 207, 169, 129, 226, 196, 39, 47, 108, 122, 159, 82, 225, 21, 56}
B_{11}={ 252, 32, 66, 199, 8, 228, 9, 85, 94, 140, 20, 118, 96, 255, 223, 215}
B_{12}={ 250, 11, 33, 0, 26, 249, 166, 185, 232, 158, 98, 76, 217, 145, 80, 210}
B_{13}={ 24, 180, 7, 132, 234, 91, 164, 200, 14, 203, 72, 105, 75, 78, 156, 53}
B_{14}={ 69, 77, 84, 229, 37, 60, 12, 74, 139, 63, 204, 167, 219, 107, 174, 244}
B_{15}={ 45, 243, 124, 109, 157, 181, 38, 116, 242, 147, 83, 176, 240, 17, 237, 131}
B_{16}={ 103, 9, 148, 235, 38, 168, 107, 189, 24, 52, 27, 187, 191, 114, 247, 64}
B_{17}={ 72, 156, 81, 47, 59, 85, 227, 192, 159, 216, 211, 243, 141, 177, 255, 167}
B_{18}={ 220, 134, 119, 215, 166, 17, 251, 244, 186, 146, 145, 100, 131, 241, 51, 239}
B_{19}={ 44, 181, 178, 43, 136, 209, 153, 203, 140, 132, 29, 20, 129, 151, 113, 202}
B_{20}={ 163, 139, 87, 60, 130, 196, 82, 92, 28, 232, 160, 4, 180, 133, 74, 246}
B_{21}={ 84, 182, 223, 12, 26, 142, 222, 224, 57, 252, 32, 155, 36, 78, 169, 152}

密钥生成器如图 4-38、4-39 所示。

5．例子

下面考虑 128 比特明文分组，128 比特密钥的例子。

设 128 比特用户密钥为：

187，183，41，209，77，47，126，223，62，75，148，150，109，3，219，183

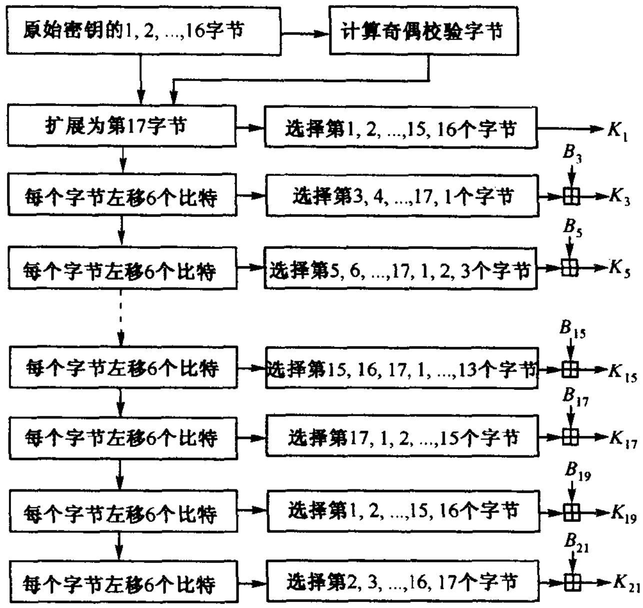


图 4-38　Safer++128 子密钥 $K_1,K_3,\ldots,K_{21}$ 的生成

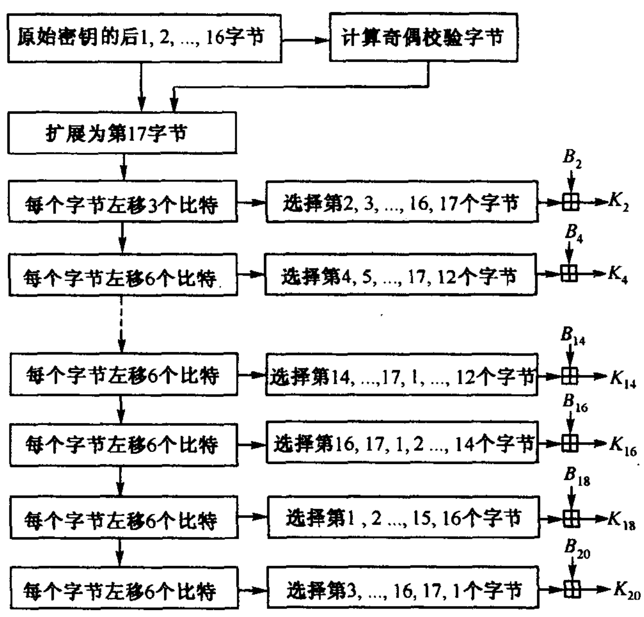


图 4-39　Safer++128 子密钥 $K_2,K_4,\ldots,K_{20}$ 的生成

128 比特明文分组为：

100，73，199，247，219，132，118，11，126，64，147，2，241，36，122，215

则输出的密文为：

73，209，104，215，224，239，100，92，200，9，126，183，19，210，64，160

15 个轮子密钥 K_1，K_2，…，K_{15} 为：

K_1= {187, 183, 41, 209, 77, 47, 126, 223, 62, 75, 148, 150, 109, 3, 219, 183}
K_2= { 3, 224, 63, 36, 28, 170, 14, 251, 31, 219, 103, 52, 114 6, 105, 118}
K_3= { 54, 31, 253, 145, 6, 140, 231, 223, 29, 63, 81, 46, 92 201, 149, 43}
K_4= { 45, 93, 54, 133, 41, 101, 204, 114, 112, 153, 241, 139, 5 31, 223, 15}
K_5= { 49, 73, 121, 28, 184, 37, 165, 36, 248, 241, 123, 246, 224 84, 222, 38}
K_6= {193, 160, 167, 83, 215, 99, 72, 177, 152, 45, 193, 114, 250 28, 216, 119}
K_7= {214, 131, 120, 11, 57, 139, 139, 139, 112, 128, 0, 39, 184 19, 106, 255}
K_8= { 53, 151, 133, 99, 2, 235, 114, 28, 195, 87, 87, 158, 212 188, 2, 17}
K_9= {187, 248, 70, 133, 47, 138, 169, 44, 72, 206, 185, 185, 32 123, 161, 177}
K_{10}= { 26, 49, 131, 20, 153, 192, 129, 57, 12, 41, 195, 45, 188 90, 8, 54}
K_{11}= { 33, 197, 157, 135, 254, 209, 153, 67, 75, 214, 136, 201, 43 158, 214, 102}
K_{12}= { 39, 229, 39, 183, 137, 125, 29, 40, 58, 65, 252, 170, 213 80, 204, 104}
K_{13}= {238, 228, 196, 255, 14, 22, 31, 90, 43, 159, 58, 80, 72 49, 80, 126}
K_{14}= {198, 58, 47, 6, 2, 23, 160, 50, 49, 214, 11, 150, 250 16, 248, 63}
K_{15}= {156, 209, 133, 91, 123, 89, 109, 169, 174, 140, 210, 168, 29 99, 71, 56}

6. Safer++分析

（1）加密结构

Safer++没有采用 Feistel 网络结构。Feistel 网络结构的优点是加、解密运算相似，但是其扩散能力较弱。例如，Feistel 网络结构需要两轮才能改变输入的每一个比特。并且 Feistel 网络结构的密码强度基于一个看似随机的 S-盒，它的安全性分析是困难的。

Feistel 网络结构的一般化是一种称为“代换——置换”的分组迭代算法。这种结构的一轮中，首先是一个由轮密钥控制的可逆函数，其次是一个坐标置换。“代换——置换”结构比 Feistel 网络结构自由度更多，所以其快速扩散能力更强。但是，却没有了加、解密过程的相似性。

Safer++既没有采用 Feistel 网络结构，也没有采用“代换——置换”结构。它采用一种可称为“代换/线性——变换”的方法：首先应用一个密钥控制的可逆函数，其次采用一个可逆线性变换。因为可逆坐标置换也是一个可逆的线性变换，所以，“代换/线性——变换”的方法可视为“代换——置换”结构的一般化。通过恰当的线性变换的选择，可以使扩散速度更快。

Safer++的操作都是基于字节的，这对软件和硬件的速度都是非常有益的。即使在 8-比特微处理器上运行都非常优秀。

（2）两种群运算和指数、对数运算的性质

Safer++使用了两种群运算，一种为逐比特模 2 加法，即字节的异或运算 $\oplus$，一种为模 256 加法运算$\boxplus$，

指数运算为：$\exp(x)=45^x \bmod 257$，（离散）对数运算：$\log(x)=\log_{45}(x)$。

且有性质：

$\exp(x_1 \boxplus x_2) = \exp(x_1) \boxtimes \exp(x_2)$，

$\exp(x_1 \oplus x_2) \neq \exp(x_1) \boxtimes \exp(x_2)$。

其中 $A \boxtimes B = AB \bmod 257$。

不等的性质是 Safer++能够抗线性分析的本质。

使用指数、对数函数还有两个理由：指数及其逆函数是典型的数学函数，它们的使用消除了对类似于“似乎随机的 S-Box”中人为弱性的怀疑；另外，研究表明，就其中一个比特

而言，其分布具有随机布尔函数的特征。

（3）矩阵 M 产生的扩散

Safer++和 Safer 族以前的版本不同之处为加密轮中可逆非线性变换的选取。Safer+选取的可逆非线性变换已较以前版本具有更好的扩散功能。而 Safer++的可逆非线性变换的扩散性更强，而且运行速度更快。

Safer++的可逆非线性变换由 16 个字节的预置乱置换、4 个 4-PHT、16 个字节的中间混乱置换及 4 个 4-PHT 接联复合构成。两个置乱置换采用“美国置乱”置换方法，对于第 i 个输入字节，输出为第 j 个输出字节，i，j 模 4 同余。可逆线性变换等价于线性变换：

$$y=xM$$

$$M=\begin{pmatrix}
1&2&2&1&4&1&1&1&1&1&1&2&1&2&2&1\\
2&1&2&1&2&1&1&2&1&1&2&1&1&4&1&1\\
1&1&4&1&2&2&1&1&2&1&1&1&1&2&1&2\\
1&1&2&1&2&1&1&1&1&1&1&1&1&2&1&1\\
1&1&2&1&1&2&1&1&4&2&1&1&1&1&2&2\\
1&1&1&2&1&1&2&1&2&4&1&1&2&1&2&1\\
1&2&1&1&2&1&1&1&2&2&1&2&1&1&4&1\\
1&1&1&1&1&1&1&1&2&2&1&1&1&1&2&1\\
4&1&1&1&1&2&2&2&1&1&2&1&1&2&1&1\\
2&1&2&1&1&4&2&1&2&1&1&1&1&1&1&2\\
2&1&1&2&1&2&4&1&1&2&1&1&2&1&1&1\\
2&1&1&1&1&2&2&1&1&1&1&1&1&1&1&1\\
1&2&1&2&1&1&2&1&1&2&2&1&4&1&1&1\\
1&4&1&1&2&1&1&1&1&1&2&2&2&1&2&1\\
2&2&1&1&1&1&1&2&1&1&4&1&2&2&1&1\\
1&2&1&1&1&1&1&1&1&1&2&1&2&1&1&1
\end{pmatrix}$$

其中 $x=(x_1,x_2,\cdots,x_{16})$，$y=(y_1,y_2,\cdots,y_{16})$。

可见矩阵 M 的每一行至少有 10 个 1，这意味着，如果输入只有一个非 0 字节，则输出至少有 10 个非 0 字节，所以由 M 产生的扩散是非常快的。Safer++还能够高效地抗击差分分析。

（4）尺度可调性

Safer++和其以前的版本一样，能够调整分组大小。例如，Safer++标准版可降为“微型版本”，这种版本便于算法性质的研究。Safer++标准版中的有关大小可相应地降为：

字节——→2 比特，

明文分组大小——→32 比特，

模数 256，257——→模数 4，5；

同样也可以构造一个小型版本，相应尺度为：

字节——→4 比特，

明文分组大小——→64 比特，

模数 256，257——→模数 16，17；

也可以构造一个巨型版本，相应尺度为：

字节——→16 比特，

明文分组大小——→256 比特，

模数 256，257——→模数 2^{16}，$2^{16}+1$；

对于任意 m，如果 2^m+1 是素数，对模 2^m+1 乘法，设 a 的阶为 2^m，则可以产生一个“m-比特/符号的 a 型 Safer++版本”，（标准版中 $a=45$）。

这样版本中对应尺度为：

字节——→m 比特，

明文分组大小——→16 m 比特，

模数 256，257——→模数 2^m，2^m+1。

这种尺度可调性使得我们可以对微型版本的脆弱性进行彻底研究，进而发现标准版本的相应特性，这在标准版本中是无法进行的。例如，通过“微型版本”的研究发现 Safer 以前版本中 PHT 层的“潜在脆弱性”事实上并不存在。

（5）优点

Safer++被确定为 NESSIE 候选算法，是因为它具有以下优点：

1）安全性研究比较充分。因为 Safer 系列早已出现，所以其安全性已得到了充分的研究。

2）Safer++的操作基于字节操作，所以非常便于实现。

3）透明性：Safer++的非线性层基于常用数学函数，而不是“似乎随机的 S-Box”。没有人为的因素，不存在可能的“后门”。

4）适用性好：Safer++具有和 DES 同样的适用性，可用于 ECB，CBC，CFB，OFB 模式。也可用于 Hash 函数，消息认证码（MAC），伪随机数的生成等。

5）环境的适应性：Safer++可在 8-比特以上平台上实现。

（6）安全性

对 Safer++最有效的攻击方法似乎只有 Biham 和 Shamir 提出的差分密码分析。研究表明，6 轮 Safer++对差分密码分析是免疫的。对于 128 比特密钥，轮数 7 的选择对安全性已给出了保险的余地。对于 256 比特的密钥，10 轮的选择提供了超级安全性。

目前还没有发现 Safer++的任何密码学脆弱性，对于 128 比特密钥，256 比特密钥，选择明文攻击并不比密钥穷举攻击更有效。且已经确定，Safer++没有“弱密钥”。

但是对目前已知的某些攻击，还没有证明 Safer++是免疫的，且加、解密过程不同使得它的实现成本大大地提高。

4.5 习题

1．证明 DES 的对称性：$\overline{DES_k(x)}=DES_{\bar{k}}(\bar{x})$，“$\overline{\ \ }$”表示按位取补运算。

2．寻找 PC_1，PC_2 的变换规律，写出其代数表达式。

3．设 $GP(2^8)=Z_2[x]/(m(x))$，对 $m(x)=x^8+x^4+x^3+x+1$，$m(x)=x^8+x^7+x^3+x+1$，

分别计算 $a(x)=x^6+x^4+x+1$，$b(x)=x^7+x^6+x^3+1$ 的乘积 $a(x)b(x)$。

4. 在有限域 $GP(2^8)=Z_2[x]/(m(x))$，$m(x)=x^8+x^4+x^3+x+1$，中计算下列元素的乘积：

（1）B7*3F；（2）9D*57；（3）11*FF；（4）DA*AD；

其中 BF="10111111" $=x^7+x^5+x^4+x^3+x^2+x+1$，11="00010001" $=x^4+1$，3F，9D，57，FF，DA，AD 类似。

5. 设 $A(x)=1bx^3+03x^2+ddx+a1$，$B(x)=acx^3+f0x+2d$ 为系数在 $GP(2^8)$ 中的两个多项式，计算 $A(x)B(x) \bmod x^4+1$，其中 $1b$，ac 表示 $GP(2^8)=Z_2[x]/(x^8+x^7+x^3+x+1)$ 中元 "00011011" $=x^4+x^3+x+1$，"10101100" $=x^7+x^5+x^3+x^2$。

4.6 实验题

1. 对于一个 S-盒 S_i，A，Δ 是两个 6 比特的输入，研究 $S_i(\Delta)$ 和 $S_i(A)\oplus S_i(A\oplus\Delta)$ 的关系，对 8 个 S-盒，写出 $S_i(\Delta)$ 和 $S_i(A)\oplus S_i(A\oplus\Delta)$ 所有可能的值，并给出 $S_i(A)\oplus S_i(A\oplus\Delta)$ 出现的概率。

2. 完整实现 DES 加解密过程。

3. 对 DES 密文的雪崩效应进行仿真检验。

4. 实现“低——高位”算法，并分析其时间，空间复杂性。

5. 完整实现 IDEA 加解密过程。

6. 对 IDEA 密文的雪崩效应进行仿真检验。

7. 用插值方法求 AES 的 S-盒变换的代数多项式表达式：

$S(x)=99+143x^{127}+181x^{191}+x^{223}+244x^{239}+37x^{247}+249x^{251}+9x^{253}+5x^{254}$。

8. 用变换 t：$x\mapsto 7^x$ 替换 AES 中的倒数变换 t：$x\mapsto x^{-1}$，计算 AES 的 S-盒。

9. 用 $a(x)=$"00000010"，$b(x)=$"01101101" 替换 AES 仿射变换：

$L_{u,v}$：$u(x)\mapsto a(x)u(x)+b(x) \bmod x^8+1$，中的参数，计算 AES 的 S-盒。

10. 改变有限域 $GP(2^8)=Z_2[x]/(m(x))$ 中 $m(x)$ 为 $m(x)=x^8+x^7+x^3+x+1$，计算计算 AES 的 S-盒。

11. 对 AES 密文的雪崩效应进行仿真检验。

第5章　公钥密码算法

5.1　引言

对称密钥密码体制的一个薄弱环节是密钥的传递。几乎所有保密通信都为此付出过沉重的代价。

（1）用于传递秘密密钥的安全信道又成为保密战线中的保密战线；

（2）由于秘密密钥传递的困难性，使得秘密密钥不能及时更换，一旦泄密，整个保密系统将完全崩溃；

（3）由于秘密密钥传递的巨大代价，使得保密通信的范围受到很大的限制，所以，保密通信常常局限于军事和国家机要用途；

（4）对称密钥是点对点的，因此，有多少通信对象，就需要多少秘密密钥，这给本来就困难的密钥保密增加了更沉重的负担。

随着计算机和通信技术的飞速发展，保密通信的需求越来越广泛，从网上政府，VPN，电子商务，移动通信，到电子邮件，网上聊天都需要保密通信。这些需求，需要数量巨大的密钥，自然，密钥分配就成了瓶颈。特别地，象电子商务，移动通信，网上聊天等保密需求，往往通信的双方互不相识，更谈不上密钥互递。有时候，通信的内容是意向性的，暂时谈不上什么价值，从而也不可能化巨大的代价传递密钥，但它们却有保密的必要。总之，将保密通信应用到普通老百姓和一般的信息传递中，密钥分配成了关键的障碍。如何（几乎）无代价地实现密钥分配成了密码学研究核心内容。

那么，如何实现这种无接触保密通信呢？也就是说，没有任何接触的通信双方如何传递、分配密钥呢？

1976年，Diffie和Hellman在美国国家计算机大会提出了他们革命性的思想——公开密钥密码思想，1978年发表了“New directions in cryptogyaphy”的著名论文，奠定了公钥密码的里程碑。

5.1.1　公钥密码思想

事实上，现实生活中到处都体现出公钥密码体制的思想，可以说，公钥密码思想是现实生活的提炼和抽象，学习应用密码学知识也需要结合现实生活，这样才能把握本质，才能创新。这也是一般科学研究的基本方法。下面以两个生活中的常识性的例子来说明公钥密码思想。

1．邮箱的科学

过去人们经常给远方的朋友寄信，所以对邮箱并不陌生。但是，可能谁都没有注意到邮箱的设计中体现的密码学思想，即使Diffie和Hellman的公钥思想已提出了二十多年。其实，稍加指点，大家就会恍然大悟。

邮箱有一个入口，用于投入信件，一个出口，是信差取信的地方。假设信差是忠于职守

的，他会把取出的信件安全地送到收信人的手中。或者干脆认为取信人就是收信人的朋友。那么，投信过程就是加密过程，除了信差以外的任何人都不能获取信件。而信差有信箱的钥匙，所以他可以打开信箱获取信件。打开信箱的过程就象解密过程。信箱就是加密算法，投入口就象加密密钥，信箱钥匙就是解密密钥。

在这里可以看到，投入口不是取信口，也就是说，加密密钥与解密密钥不同，这就是公钥密码的本质，也就是投入口的单向操作性。由此可以马上想到现实生活中有许多单向操作的现象，如投票箱和锁等。

2．保密信盒

设 A，B 两个人需要进行保密通信，他们选择一个牢固的信盒。

（1）A 把需要发送的信件装入信盒，用自己的锁锁上信盒，钥匙自己保管好，把信盒送给 B；

（2）B 收到信盒后，由于没有钥匙，当然不能打开信盒。相反地，他在信盒的另一个并排的锁环处又加上了自己的锁（并列），钥匙也自己保管起来，把信盒又回送给 A；

（3）A 收到信盒后，打开自己的锁，把信盒再次送给 B；

（4）B 收到信盒，用自己的钥匙打开信盒，取出信件。

这样 A，B 就实现了一次保密通信。

我们看到：

（1）这是一次无接触信息交换，不需要密钥的传递过程；

（2）这是一个两趟协议，从而是一种非实时通信方式，不便用于直接通信，一般用来传递对称密钥，所以这一协议称为密钥交换协议；

（3）信盒的设计需要：两把锁可以并列地锁上、打开，互不影响。

5.1.2 Diffie-Hellman 密钥交换协议

1976 年，Diffie 和 Hellman 提出公钥密码思想时利用幂模的方法实现了上述信盒的思想，给出了一种无接触密钥交换算法，

设 A，B 是通信的双方，他们共同选择一个有限群 H（称作乘法群），一个元素 $G \in H$；

（1）A 随机地选择一个整数 k_A，计算 $Q_A = a^{k_A} \in G$，将 k_A 保密，Q_A 公开；

（2）B 随机地选择一个整数 k_B，计算 $Q_B = a^{k_B} \in G$，将 k_B 保密，Q_B 公开；

（3）A 计算得公钥：$k = Q_B^{k_A} = G^{k_A k_B}$；

（4）B 计算得公钥：$k = Q_A^{k_b} = G^{k_A k_B}$。

5.2 RSA 密码体制

1978 年，Ronald.L.Rivest，Adi Shamir 和 Leonard M.Adleman 发表了著名的论文“A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”（获得数字签名和公开密钥密码系统的一种方法），继 MH 背包公钥密码体制以后，提出了第一个有效的公钥密码体制，现被用他们的名字的首字母命名，称为 RSA 公钥密码体制。这是一个容易理解的方法，得到了广泛的应用，先后被 ISO，ITU，SWIFT 等国际化标准组织采用作为标准，RSA 已经成

为事实上的国际标准。RSA 在美国申请了专利，专利期限到 2000 年 9 月 20 日为止，在其他国家没有专利。

5.2.1 体制描述

设 $n=pq$，p 和 q 是两个奇素数，设 $P=C=Z_n$，

$K=\{(n,p,q,a,b)|\ n=pq,\ p,q$ 是素数，$ab\equiv1 \bmod \varphi(n)\}$。

对于 $\forall\ k=(n,p,q,a,b)$，定义：

加密算法：$E_k(m)=\left(m^a \bmod n\right)$，$\forall m\in P$，

解密算法：$D_k(c)=\left(c^b \bmod n\right)$，$\forall c\in C$。

其中：n，a 公开，p,q,b 是保密的。

5.2.2 算法分析

RSA 密码体制的原理基于 Euler 定理及大数分解的困难性，体制中运算在 Z_n 中进行。

（1）设 n 是两个不同奇素数 p，q 的乘积，对于这样的正整数 n，其 Euler 函数值是容易计算的，它是 $\varphi(n)=(p-1)(q-1)$。但是，如果不知道 p,q，计算 $\varphi(n)$ 是困难的。

（2）对于给定的明文 $0<x<n$，选定一个正整数 a，称为加密密钥，作 Z_n 中的指数运算 $x^a \bmod n$，将明文以指数形式表示出来，或者说以指数形式将明文隐藏起来。

（3）如果仅知道 x^a，将无法得到任何其他信息。

（4）如果攻击者设法得到了一个明、密文对 (m,c)，他想得到解密密钥 b，则必须在 Z_n 中求解（离散）对数问题：$b=\log_c m$，这是一个困难问题。可见这种加密思想是简单、明确的。

（5）合法的解密过程的基本依据是 Euler 定理：

$(x,n)=1$ 时，$x^{\varphi(n)}\equiv1 \bmod n$。

如果用 a 表示 b 模 $\varphi(n)$ 的逆元，即：$ab\equiv1 \bmod \varphi(n)$，那么，$(x^b)^a=x^{ab}\equiv x \bmod n$。这就还原出了明文 x。从此可见，解密的关键是求 b 模 $\varphi(n)$ 的逆元，归根结底是计算 $\varphi(n)$。对于加密者来说，这是已知的，从而解密过程就可以顺利地完成了。但对分析者来说，一般地，只有分解 n 才能求出 $\varphi(n)$。

（6）解密原理的严格叙述如下：

如果 $ab\equiv1 \bmod \varphi(n)$，$x<n$，则：$\left(x^b\right)^a\equiv x \bmod n$。

证明 因为 $ab\equiv1 \bmod \varphi(n)$，所以，存在某整数 $k\geqslant1$，$ab=k\varphi(n)+1$。

1）若 $(x,n)=1$，则由 Euler 定理有 $x^{\varphi(n)}\equiv1 \bmod n$，于是：

$$\begin{aligned}(x^b)^a &\equiv x^{k\varphi(n)+1} \bmod n\\ &\equiv x\left(x^{\varphi(n)}\right)^k \bmod n\\ &\equiv x\cdot1^k \bmod n\\ &\equiv x \bmod n\end{aligned}$$

2）若 $(x,n)>1$，则由于 $n=pq$， $x<n$，所以 (x, n) 必为 p，q 之一。

不失一般性，设 $(x, n)=p$，即有：$x=cp, 1\leqslant c<q$。这时 $(x,q)=1$，所以，对于任意 k，根据 Euler 定理：$x^{\varphi(q)}\equiv 1 \mod q$，

$$\Rightarrow \quad x^{k(p-1)\varphi(q)}\equiv 1^{k(p-1)}\equiv 1 \mod q,$$

$$\Rightarrow \quad x^{k(p-1)(q-1)}\equiv 1 \bmod q,$$

$$\Rightarrow \quad x^{k\varphi(n)}\equiv 1 \mod q,$$

$$\Rightarrow \quad 1=x^{k\varphi(n)}+sq, \quad s\in Z。$$

两边同乘 $x=cp$，得：

$$x=x^{k\varphi(n)+1}+cs\,pq,$$

$$\Rightarrow x^{k\varphi(n)+1}\equiv x \mod pq。$$

可见，加密、解密过程中的主要运算是 Z_n 中的幂运算。由于 n 可能非常大，所以模 n 的幂运算的效率就成为算法效率的关键。

对于加密算法：$x^b \mod n$，在 1.5 中已经给出了一种高效的“平方—乘法”运算，可以使模 n 的幂运算快速进行。

设 $b=\sum_{k=0}^{l-1} b_k 2^k$ 是加密密钥的二进制形式。进一步写成：

$$x^b=x^{b_0}x^{b_1 2}x^{b_2 2^2}\cdots x^{b_{l-1}2^{l-1}}$$

$$=x^{b_0}\left(x^{b_1}\left(x^{b_2}\cdots\left(x^{b_{l-2}}\left(x^{b_{l-1}}\right)^2\right)^2\cdots\right)^2\right)^2。$$

于是，我们可用下列过程来计算：

a）令 $z=1$；

b）i 从 $l-1$ 递减到 0；

先计算：$z:=z^2 \mod n$；

如果 $b_i=1$，则计算：$z:=z\cdot x \mod n$；

可见，需要进行：

b 的二进制表示，这是计算机的基本数制，从而运算时间可以忽略；

l 次平方运算；

最多 l 次乘法，l 次判断（时间可以忽略）。

如果将平方运算也看作乘法运算，则共需最多 $2l=\log_2 b<\log_2\varphi(n)$ 次乘法。

另外，加密密钥 b 必须对模 $\varphi(n)$ 有逆元，所以 b 的选取必须满足：（b，$\varphi(n)$）$=1$。

5.2.3 安全性分析

1．RSA 的安全性与因子分解的关系

定理 5-1 因子分解 n 与求 $\varphi(n)$ 等价

证明 如果已经分解了 $n=pq$，则 $\varphi(n)=(p-1)(q-1)$；

反之，如果已知 $\varphi(n)$，则：$\phi(n)=(p-1)(q-1)=pq-(p+q)+1=n+(p+q)+1$，

$p+q=n-\varphi(n)+1, (p-q)^2=p^2-2pq+q^2=(p+q)^2-4pq$

$p-q=\sqrt{(p+q)^2-4n}=\sqrt{(n-\varphi(n)+1)^2-4n}$

得到：$p=\dfrac{n-\varphi(n)+1+\sqrt{(n-\varphi(n)+1)^2-4n}}{2}$，

$q=\dfrac{n-\varphi(n)+1-\sqrt{(n-\varphi(n)+1)^2-4n}}{2}$。定理证毕。

定理 5-2 RSA-n 是安全的，则 n 是分解困难的。

证明 反设，如果已经分解的 $n=pq$，则可计算出 $\varphi(n)$，从而，由加密密钥 e，求解逆元 $d=\left(e^{-1} \bmod \varphi(n)\right)$，得到解密密钥，从而攻破 RSA-$n$，矛盾。定理证毕。

但是，目前还没有证明：n 是分解困难的可保证 RSA-n 是安全的。

大多数密码学家都认为这是正确的。人们对 RSA 的信心也是建立在这个没有证明的猜想上的。

对这一猜测的担心就是：不通过分解因式是否能够攻破 RSA？

对这一担心有利的支持是下面例子：

设：$p=17$，$q=11$，$n=pq=187$，$b=7$，明文为 123。

$c_1=123^7 \equiv 183 \mod 187$

得密文 183，但是，

$c_2=183^7 \equiv 72 \mod 187$

$c_3=72^7 \equiv 30 \mod 187$

$c_4=30^7 \equiv 123 \mod 187$

可见，只需对密文连续加密 4 次即可恢复明文。这种攻击方式称为迭代攻击。

作为练习，请讨论迭代攻击成功的一般条件。

2．关于因子分解

大数分解是一个 NP 问题，目前已知的最好的算法需要进行 $e^{\sqrt{\ln n \ln(\ln n)}}$ 次算术运算。假设用一台每秒运算 10^6 次的计算机来分解一个 160 位十进制的数，则有如下的计算：

$n=10^{160}$，$\sqrt{\ln n \ln \ln n}\approx 46.65868$，

1 年 $=365\times24\times60\times60=3153600\approx e^{17.2666}$ 秒；

$$\frac{e^{\sqrt{\ln n \ln\ln n}}/10^6}{3153600}\approx e^{46.65868}/e^{31.0821}\approx e^{15.57648}\approx 5.81834\times10^7 \text{ 年。}$$

即，要分解一个 160 位十进制的数，需要 5.81834×10^7 MIPS 年。类似地，可计算出分解一个 200 位的十进制整数需要 3.80528×10^{10} MIPS 年。可见，增加 n 的位数，将大大地提高体制的安全性。

直接分解一个大合数的 RSA 攻击的一个实例是：1994 年 4 月贝尔实验室 Lenstra 领导的小组对 RSA-129 的攻击，他们分解了 Rivest，Shamir，Adleman 提出 RSA 算法时给出的一个很难分解的 129 位十进制的合数。分解时启用了 1600 台计算机，耗时 8 个月，用时 4600MIPS

年。Pentium100 分解 RSA-129 需要 37 年，100 台 Pentium100 需要 4 个月。

商业应用中，一般认为 RSA-155 是安全的（模的位数约 512 比特）。

3．强素数

由前面讨论知道，素数 p,q 的选取是 RSA 安全的关键，p,q 的选择必须使得分解 $N=pq$ 是计算上不可能的。一般地认为 p,q 必须是强素数。

所谓强素数满足：

（1）存在两个大素数 p_1，p_2，使得 $p_1 \mid p-1$，$p_2 \mid p+1$；

（2）存在 4 个大素数 r_1,s_1,r_2,s_2，使得 $r_1 \mid p_1-1$，$s_1 \mid p_1+1$，$r_2 \mid p_2-1$，$s_2 \mid p_2+1$。

p,q 称为 1 级素数；p_1,q_1 称为 2 级素数，r_1,s_1,r_2,s_2 称为 3 级素数。

强素数的条件是必要的，否则有下面的“p–1 方法”。

p–1 因子分解法：

如果 $N=pq$，$p-1=\prod_{i=1}^{t} p_i^{a_i}$，$0 \leqslant p_i < B$ 为第 i 个素数，B 是一个已知的小数。

令：$R=\prod_{i=1}^{t} p_i^{a}$，$a \geqslant a_i, i=1,2,\cdots,t$。那么 $p-1 \mid R$，由费尔马定理可知，对 $\forall b$，若 $(b,p)=1$，则 $b^R \equiv 1 \mod p$。

考虑 $X=\left(2^R \mod N\right)$，如果 $X=1$，则用 3 取代 2，考虑 $X=\left(3^R \mod N\right)$；如果仍有 $X=1$，则用 4 代替 2，如此继续，直到 $X \neq 1$。

当 $X \neq 1$ 时，有 $b^R = X + lN$，所以，$X-1=b^R-1-lN$，由于 $b^R \equiv 1 \mod p$，所以 $p \mid \left(b^R-1-lN\right)$，再由于 $X<N$，所以，$GCD(X-1,N)=p$。这样就得到了 N 的因子。

其中，B 为小素因子的界，a 为素因子方幂的估计。

【例 5-1】 $N=150463=pq$，应用“$p-1$ 方法”分解 N。

解 取 $B=8$，$a=3$，则：$R=\prod_{p_i<8} p_i^3 = 2^3 \times 3^3 \times 5^3 \times 7^3 = 9261000$。

$2^R \mod N = 139473$，$GCD(139473-1,150463)=379$，所以得到 N 的分解式 $150463=379 \times 397$。

如果 B 取得过小，或 a 取得过小，就不能保证 $p-1 \mid R$，从而不能保证 $b^R \equiv 1 \mod p$，也就不能保证 $p \mid (X-1)$，这样，必有 $GCD(X-1,N)=1$，因而计算不成功。只须加大 B，a 重新计算即可。

例如，若例题中取 $a=2$，则 $R=44100$，$2^R \mod N=23549$，则得：$GCD(23549-1,150463)=1$。

Williams 也提出了一个类似于“$p-1$ 方法”的“$p+1$ 方法”，所以，要保证 RSA 安全，素数首先必须抗“$p-1$、$p+1$ 方法”的分析，这就是强素数的条件。

4．素因子之差

RSA 的素数不但要在强素数中选取，而且 p,q 的差必须很大。否则，首先可估计 p,q 的

平均值：$\tilde{Q}=\frac{p+q}{2}=\left[\sqrt{pq}\right]=\left[\sqrt{N}\right]$、$\left[\sqrt{N}\right]+1$ 或 $\left[\sqrt{N}\right]+2$ 等，使得：$\tilde{Q}^2-N$ 为完全平方数。

于是，$\left(\frac{p-q}{2}\right)^2=\left(\frac{p+q}{2}\right)^2-N=\tilde{Q}^2-N$，这样可得到：

$$\begin{cases}\frac{p+q}{2}=\tilde{Q}\\ \frac{p-q}{2}=\sqrt{\tilde{Q}^2-N}\end{cases}$$，求出 p,q，就得到 N 的分解式。

【例 5-2】 $p=101,q=103,N=101\times103=10403$。

解 $\sqrt{N}=101.995$，令 $\frac{p+q}{2}=102$，那么，$102^2-10403=1^2$，（如果取 $\frac{p+q}{2}=101$，则 $101^2-10403<0$，不合适）所以得到：

$$\begin{cases}\frac{p+q}{2}=102\\ \frac{p-q}{2}=1\end{cases}$$，解得 $p=101,q=103$。

5. 低指数攻击

（1）小加密密钥攻击

为了提高加密速度，一般尽量选取较小的加密密钥 e。但是如果 e 选取过小，则系统是不安全的。

例如：对 $RSA-n$，设明文为 m，$0\leqslant m<n$。如果 e 过小，则可能有：$m^e<n$。那么 $c=\left(m^e \mod n\right)=m^e$，这就是普通的指数，从而 $m=\sqrt[e]{c}$ 称为普通开方；

（2）小解密密钥 d 攻击

1990 年，Micgael Wiener 证明了，若 d 的长度小于模 n 长度的 $\frac{1}{4}$ 时，利用连分数可在多项式时间内计算出 d（关于连分数在密码学中的应用请看有关著作）。

6. 共模攻击

为了密钥管理的方便，通常认为，对一个网络用户群，可以选取一个共同的 RSA-n 密码体制，对每个用户或每一种业务分发不同的密钥对 (e_i,d_i)。

如果将同一消息 m，用两个互素的公钥 e_1,e_2 加密，即：

$$\begin{cases}c_1=\left(m^{e_1} \mod n\right)\\ c_2=\left(m^{e_2} \mod n\right)\end{cases}$$

那么，存在"共模攻击"：

由于 e_1,e_2 互素，所以存在 r,s，使得 $re_1+se_2=1$，且 r,s 中必有一个是负数，不妨设 $r<0$。

计算 $c_1^{-1} \mod n$，

则：$\left(c_1^{-1}\right)^{-r}c_2^s \mod n=\left(m^{e_1}\right)^r\left(m^{e_2}\right)^s \mod n=m^{re_1+se_2} \mod n=m$，恢复出了明文。

7. 共指攻击

为了提高加密速度，加密密钥一般比较小或满足某些特别条件，最常用的有三个：3，17，65537（即：$2^{16}+1$）。所以 RSA 体制中常常会出现不同模数，但有共同加密密钥的情形。

若有 e 个 RSA 密码体制 RSA-n_1，RSA-n_2，…，RSA-n_e，都采用公钥 e。

易见 $(n_i, n_j)=1$，$i \neq j$（否则，可以求（n_i,n_j），而分解 n_i, n_j）。如果这 e 组密钥被用作加密同一个信息 m，则：

$c_1 = (m^e \bmod n_1)$，

$c_2 = (m^e \bmod n_2)$，

……

$c_e = (m^e \bmod n_e)$，

则由孙子定理可得到：$c = (m^e \bmod n_1 n_2 \cdots n_e)$。

由于 $m < n_1$，$m < n_2$，…，$m < n_e$，所以 $m^e < n_1 n_2 \cdots n_e$，于是得到 $m^e = c$，从而 $m = \sqrt[e]{c}$ 是普通开方。这样就恢复了明文。

5.2.4 RSA 举例

下面用一个具体的例子来说明 RSA 的实现过程。

【例 5-3】 设 p=101，q=113，则 $n=pq$=11413，$\varphi(n)$=101×112=11200。

分解 $11200 = 2^5 5^2 7$，所以，b 可用作是加密指数当且仅当它不能被 2,5,7 整除（当然，实际工作中，不可能分解 $\varphi(n)$，而是，先选取 b，再验证 $\gcd(\varphi(n), b)=1$。

假设选取 $b = 3533$，则可求得其逆为：$b^{-1} \equiv 6597 \bmod 11200$，所以，解密指数是 $a = 6597$。

用户 B 在公开库中公开注册公开密钥：$n = 1143$，$b = 3533$；

但保密秘密密钥：$a = 6597$；

销毁（保密）：$\varphi(n)$=11200，$p = 101$，$q = 113$。

如果用户 A 想给 B 发送明文：$m = 9726$，他用 B 的公开密钥 $b = 3533$ 加密明文 m：

$$m^b = 9726^{3533} \equiv 5761 \mod 11413,$$

得到密文：$c = 5761$，在一个（不安全的）信道上发送给 B。

B 接到 A 发来的密文后就用他自己的解密密钥 $a = 6597$ 解密：

$$c^a = 5761^{6597} \equiv 9726 \mod 11413,$$

解得明文：$m = 9726$。

幂模运算 $9726^{3533} \mod 11413$ 的"平方—乘法"过程：

$$b = 3533 = (110111001101)_2。$$

计算过程见表 5-1。

表 5-1 平方——乘法计算过程例表

步骤 k	b_k	z
11	1	$1^2 \times 9726 = 9726$

（续）

步骤 k	b_k	z
10	1	$9726^2 \times 9726 \equiv 2659$
9	0	$2659^2 \equiv 5634$
8	1	$5634^2 \times 9726 \equiv 9167$
7	1	$9167^2 \times 9726 \equiv 4958$
6	1	$4958^2 \times 9726 \equiv 7783$
5	0	$7783^2 \equiv 6298$
4	0	$6298^2 \equiv 4629$
3	1	$4629^2 \times 9726 \equiv 10185$
2	1	$10185^2 \times 9726 \equiv 105$
1	0	$105^2 \equiv 11025$
0	1	$11025^2 \times 9726 \equiv 5761$

【例 5-4】 下面给出一个实用尺度的 RSA 实例：

选取两个 80 位以上的大素数（实用中应选取安全素数）：

p=15415466547688798760980354432431542490965479032432156652316894699005632543334
3711，

q=94645466174334344547683223871698035443243154247239096547903221468946909054625
43374051，

n=14590040177008628382012721265706881504951606814921964395791127401472161780935
98888486966328472761481381599942943845254687301640268003509241546021483122139
784721443261，

易见 n 的十进制位数为 166。

n 的欧拉函数值

$\varphi(n)$=14590040177008628382012721265706881504951606814921964395791127401472161780
935988790222655563738501386256677522297173949380765607537695328847007056414
41177996744725500。

选取公钥 e=17，计算得到私钥：

d=85823765747109578717721889798275773558538863617188025857594867067483304593741
11053072150331610883168386280895468925852576920945610409016968827680243771635
27496897353。

假设待加密信息为：

“The watermark is capable of carrying such information as authentication or authorization codes,or a legend essential for image interpretation.This capability is en-visaged to find application in image tagging,copyright enforcement,counterfeit protection,and controlled access”。

共 275 个字符，每一个字母用 3 位十进制 ASCII 码表示，则信息的编码为：

084, 104, 101, 032, 119, 097, 116, 101, 114, 109, 097, 114, 107, 032, 105, 115, 032, 099, 097, 112,

097, 098, 108, 101, 032, 111, 102, 032, 099, 097, 114, 114, 121, 105, 110, 103, 032, 115, 117, 099, 104, 032, 105, 110, 102, 111, 114, 109, 097, 116, 105, 111, 110, 032, 097, 115, 032, 097, 117, 116, 104, 101, 110, 116, 105, 099, 097, 116, 105, 111, 110, 032, 111, 114, 032, 097, 117, 116, 104, 111, 114, 105, 122, 097, 116, 105, 111, 110, 032, 099, 111, 100, 101, 115, 044, 111, 114, 032, 097, 032, 108, 101, 103, 101, 110, 100, 032, 101, 115, 115, 101, 110, 116, 105, 097, 108, 032, 102, 111, 114, 032, 105, 109, 097, 103, 101, 032, 105, 110, 116, 101, 114, 112, 114, 101, 116, 097, 116, 105, 111, 110, 046, 084, 104, 105, 115, 032, 099, 097, 112, 097, 098, 105, 108, 105, 116, 121, 032, 105, 115, 032, 101, 110, 045, 118, 105, 115, 097, 103, 101, 100, 032, 116, 111, 032, 102, 105, 110, 100, 032, 097, 112, 112, 108, 105, 099, 097, 116, 105, 111, 110, 032, 105, 110, 032, 105, 109, 097, 103, 101, 032, 116, 097, 103, 103, 105, 110, 103, 044, 099, 111, 112, 121, 114, 105, 103, 104, 116, 032, 101, 110, 102, 111, 114, 099, 101, 109, 101, 110, 116, 044, 099, 111, 117, 110, 116, 101, 114, 102, 101, 105, 116, 032, 112, 114, 111, 116, 101, 099, 116, 105, 111, 110, 044, 097, 110, 100, 032, 099, 111, 110, 116, 114, 111, 108, 108, 101, 100, 032, 097, 099, 099, 101, 115, 115。

共 825 个数字，根据 n 的位数 166，每组 165 位，分成 5 组，得到：

84104101032119097116101114109097114107032105115032099097112097098108101032111102032099097114114121105110103032115117099104032105110102111114109097116105111110032097，

11503209711711610410111011610509909711610511111003211111403209711711610411111410512209711610511111003209911110010111504411111403209703210810110310111010003210 1115115，

10111011610509710803210211111403210510909710310103210511011610111411211410111609711610511111004608410410511503209909711209709810510810511612103210511503210111 0045118，

10511509710310110003211611103210210511010003209711211210810509909711610511111003210511003210510909710310103211609710310310511010304409911111212111410510310411 6032101，

11010211111409910110910111011604409911111711011610111410210110511603211211411111610109911610511111004409711010003209911111011611411110810810110003209709909910 1115115。

加密后得到 5 组密文：

121584700183292300199593905539964263708926950340241446998185485539726860286587892703712867608898180134317192004369189754163853055636815783326337603745561706 1292347599，

104602166810368706534514893195535805739694419559042697900567812807624163675263473051395414031967415154939038111704473283316070573388426586444219949807586490 1076813278，

978697639039251569046027137109759573263261410414889785576594889508420201227866703799567132734879134987676333371126347048431959589427342827178130652106173268 315901320，

3757362245143357543990400837017873100182125018157523676452592166373921170392
16284937628112823076303910606338772897411781824812774659513812271140005702605018
587368208，

4126925733459671845460804971333489182236764278978972174877597284046760307670
68096000460497217362733402262499298813817541482172354976829455295447923379091099
143179177。

5.3 Rabin 公钥密码体制

1979 年，M.O.Rabin 发表了“Digital Signatures and Public-Key Functions as Intractable as Factorization”（与分解因子同样困难的数字签名和公钥函数），提出了一个与因子分解困难性等价的公钥密码体制，现称为 Rabin 密码体制。

Rabin 密码体制是 RSA 密码体制的修正方案，体制的具体描述如下。

5.3.1 体制描述

设 n 是两个不同的素数 p,q 的乘积，$p\equiv 3 \bmod 4, q\equiv 3 \bmod 4$。假设 $P=C=Z_n$，定义：

加密运算：$e_k(m)=m^2 \bmod n$；

解密运算：$d_k(c)=\sqrt{c} \bmod n$。

n 公开，p,q 保密。

5.3.2 解密算法及原理

可见加密算法就是 Z_n 中的平方运算，可以认为就是 RSA 中令加密密钥为 2。但是，由于（2，$\varphi(n)$）=2≠1，所以，Rabin 体制却不能象 RSA 一样，用 $2^{-1} \bmod \varphi(n)$ 来解密（$2^{-1} \bmod \varphi(n)$ 不存在）。所以 Rabin 体制只能称为 RSA 的修正，而不是特例。

Rabin 体制的解密算法是通过孙子定理来完成的。

解密就是解同余方程：$x^2\equiv c \bmod n$　　(5-1)

考虑同余方程组：$\begin{cases} x^2\equiv c \bmod p \\ x^2\equiv c \bmod q \end{cases}$　　(5-2)

易见方程（5-1）的解一定是方程组（5-2）的解，反之由孙子定理，方程组（5-2）有惟一解：$x^2\equiv c \bmod pq$，即方程组（5-2）的解也是方程（5-1）的解。所以方程（5-1）与方程组（5-2）等价。

从而，我们只要求解（5-2）即可。

由于 $p\equiv 3 \bmod 4$，所以 $\dfrac{p+1}{4}$ 是正整数，所以：

$$\left(\pm c^{\frac{p+1}{4}}\right)^2 = c^{\frac{p+1}{2}} = c^{\frac{p-1}{2}+1} \equiv \left(x_0^2\right)^{\frac{p-1}{2}} c = x_0^{p-1}c \equiv c \bmod p。$$

（由费尔马小定理：$x_0^{p-1} \equiv 1 \bmod p$）

所以 $x^2 \equiv c \bmod p$ 等价于 $x \equiv \pm c^{\frac{p+1}{2}} \bmod p$。

同理：$x^2 \equiv c \bmod q$ 等价于 $x \equiv \pm c^{\frac{q+1}{2}} \bmod q$。

最后，方程组（5-2）等价于：

$$\begin{cases} x \equiv \pm c^{\frac{p+1}{2}} \bmod p \\ x \equiv \pm c^{\frac{q+1}{2}} \bmod q \end{cases}，即：$$

$$\begin{cases} x \equiv c^{\frac{p+1}{2}} \bmod p \\ x \equiv c^{\frac{q+1}{2}} \bmod q \end{cases},\quad \begin{cases} x \equiv c^{\frac{p+1}{2}} \bmod p \\ x \equiv -c^{\frac{q+1}{2}} \bmod q \end{cases},\quad \begin{cases} x \equiv -c^{\frac{p+1}{2}} \bmod p \\ x \equiv c^{\frac{q+1}{2}} \bmod q \end{cases},\quad \begin{cases} x \equiv -c^{\frac{p+1}{2}} \bmod p \\ x \equiv -c^{\frac{q+1}{2}} \bmod q \end{cases}。$$

应用孙子定理来解这四个方程组，令：

$y_1 = \left(q^{-1} \bmod p\right) = q^{p-2}$，$y_2 = \left(p^{-1} \bmod q\right) = p^{q-2}$，

则四个解为：$x = \pm y_1 q c^{\frac{p+1}{4}} \pm y_2 p c^{\frac{q+1}{4}} \bmod n = \pm q^{p-1} c^{\frac{p+1}{4}} \pm p^{q-1} c^{\frac{q+1}{4}} \bmod n$。
于是得到四个可能明文。

如何从四个可能明文中确定出哪一个是真正的明文？一般是不可能的。如果被加密的信息是自然语言，那么从解密的信息是否有意义很容易确定出正确的明文，但是对于非自然语言的数字或比特串，这就无法实现了。解决的办法是通常约定在信息中加入一个标识串。

5.3.3 安全性与因子分解

和 RSA 不一样，已证明：破译 Rabin-n 与分解 n 等价。下面来证明这一结论。

定理 5-3 设 $n = pq$，$p,q \equiv 3 \bmod 4$，则 $x^2 \equiv c \bmod n$ 可解的充要条件是 n 可分解。

证明 由前面关于 Rabin 解密算法的讨论可知，如果能够分解 n，就可以求解 $x^2 \equiv c \bmod n$。

反之，如果已知 $x^2 \equiv c \bmod n$ 的四个解，$\pm x$，$\pm y$（该方程有解，则必有两组不同的解，每组有两个互反的解），则：

$x^2 - y^2 \equiv 0 \bmod pq$，

$\Rightarrow pq \mid (x-y)(x+y)$，

$\Rightarrow p \mid x-y, q \mid x+y$，或 $q \mid x-y, p \mid x+y$，

于是，$(x-y,n)$，$(x+y,n)$ 就是 n 的因子，用扩展欧几里德除法即可分解 n。定理证毕。

5.3.4 错误明文攻击

设 A 构造了密码体制 Rabin-n，为了证明自己是该体制的拥有者，例如，A 注册时，需要向管理员证实，A 请求 B 验证（当然不会通过向 B 显示解密算法来证实）。

于是 B 任选一个随机数 x，加密后送给 A，即将 $c=\left(x^2 \mod n\right)$送给 A，并要求解密，A 解密后得到四个值，他无法确定是哪一个，只得任选一个 y 给 B，如果 $y\neq\pm x$，则 B 求 $(y\pm x,n)$ 就可分解 n，如果 $y=\pm x$，则 B 再选一个 x，要求 A 重复以上过程，直到 $y\neq\pm x$ 为止。这样一般地 B 都可以分解 n，攻破体制。

可见，B 希望的并不是正确的明文，而是错误的明文，所以称为错误明文攻击。

5.3.5 示例

设 $p=151$，$q=223$，$n=pq=33673$。易见，$p,q\equiv 3 \mod 4$，所以有 Rabin 密码体制 Rabin-33673。

设信息为 $x=31280$，则密文为：

$$c=\left(x^2 \mod n\right)=\left(31280^2 \mod 33673\right)=2039。$$

解密：

$$y_1=q^{-1} \mod p=223^{-1} \mod 151=72^{-1} \mod 151=86，$$

$$y_2=p^{-1} \mod q=151^{-1} \mod 223=96。$$

$$c^{\frac{p+1}{4}}=2039^{38} \mod 33673=22778，$$

$$c^{\frac{q+1}{4}}=2039^{56} \mod 33673=31726。$$

所以可能密文为：

$$\begin{aligned}x&=\pm y_1 q\times c^{\frac{p+1}{4}}\pm y_2 p\times c^{\frac{q+1}{4}} \mod n\\&=\pm 86\times 223\times 2039^{38}\pm 96\times 151\times 2039^{56} \mod 33673\\&=\pm 86\times 223\times 22778\pm 96\times 151\times 31726 \mod 33673\\&=\pm 30328\pm 27935 \mod 33673\\&=\begin{cases}+30328+27935 \mod 33673=24590\\+30328-27935 \mod 33673=2393\\-30328+27935 \mod 33673=31280\\-30328-27935 \mod 33673=9083\end{cases}\end{aligned}$$

可见，原明文 31280 出现在可能明文中。

5.4 ElGamal 公钥密码体制

1984 年，ElGamal.T 发表了“A public-key cryptosystem and a signatures scheme based on discrete logarithms”（一个基于离散对数的公钥密码体制和数字签名方案），提出了一种安全性基于离散对数，既可用于加密又可用于数字签名的公钥密码体制。

5.4.1 体制描述

设 p 是一个素数，g 是 Z_p^* 的一个生成元。明文空间 $P=Z_p^*$，密文空间 $C=Z_p^*\times Z_p^*$，选定 $0<a<p$，计算 $b\equiv g^a \mod p$。

秘密密钥：a；

公开密钥：b；

加密算法：设明文为 m，$0 \leqslant m < p$，随机地选择一个正整数 k，$0 \leqslant k < p$，$(k, p-1) = 1$，计算：

$y_1 = \left(g^k \mod p\right)$，$y_2 = \left(mb^k \mod p\right)$。

密文 $c = (y_1, y_2)$。

解密算法：$m = \dfrac{y_2}{y_1^a} = \dfrac{mb^k}{g^{ak}} = \dfrac{mg^{ak}}{g^{ak}} = m \mod p$。

易见，ElGamal 体制的密文不是惟一的，这是一种非确定性加密方式。显然增加了系统的安全性，但是，代价是密文膨胀了两倍。

5.4.2 示例

设 $p = 293$，构造 ElGamal-293，可以验证 2,3,5,7,8,11,12,13,18,19,20,23,27,28,29,30,32,34,⋯都是生成元，例如取 $g = 30$。

选取秘密密钥 $a = 200$，

计算得到公钥 $b = 46$，

设明文为 $m = 137$，选取 $k = 79$，（k，p–1）=（79，292）=1，则：

$y_1 = \left(g^k \mod p\right) = 30^{79} \mod 293 = 48$，

$y_2 = \left(mb^k \mod 293\right) = \left(137 \times 46^{79} \mod 293\right) = 81$，

所以密文为 $(48,81)$。

解密：$\dfrac{y_2}{y_1^a} \equiv y_2 \times \left(y_1^a\right)^{-1} \equiv 81 \times \left(48^{200}\right)^{-1} \equiv 81 \times (84)^{-1} \equiv 81 \times 150 \equiv 137 \mod 293$。

5.5 MH 背包公钥密码体制

和 RSA 公钥体制一起，背包公钥体制被认为是两个著名的公钥体制之一。1978 年 Merkle 和 Hellman 首先提出了一个现在称为 MH 背包体制的密码体制，虽然它和其几个变形在 20 世纪 80 年代初被 Shamir 等人破译了，但是，它的思想和有关理论首先揭示了公钥密码算法的本质，所以仍然具有深刻的理论研究价值。

所谓背包问题，可通俗地描述为：设有长为 b 的背包和一组直径相同的圆盘共 n 个，圆盘厚度分别为：$a_1, a_2, \cdots, a_n$。试问能否从中选出一部分盘子，正好装满这个背包？如果有，应如何选取？数学表达如下：

5.5.1 0–1 背包问题

设 $b, a_1, a_2, \cdots, a_n$ 是已知的正整数，问方程 $a_1x_1 + a_2x_2 + \cdots + a_nx_n = b$ 是否有二进制解 $(x_1, x_2, \cdots, x_n)$，$x_i = 0$ 或 1？如有，则求解。

背包问题也称为子集和问题。已证明这是一个 NP 完全问题，没有有效的算法。1979年，R.Schroeppel 和 A.Shamir 提出了一种求解一般背包问题的算法，算法的时间复杂性为 $O(2^{n/2})$，空间复杂性为 $O(2^{n/4})$。一般情况下，问题不能在多项式时间内求解，但是，在某些特殊情况下，仍然可能在多项式时间内求解，例如下面的超递增序列情况。

以下假设，除特殊说明外，数据都是非负整数。

5.5.2 超递增序列

如果序列 $a_1,a_2,\cdots,a_n$ 满足条件：$a_k > a_1+\cdots+a_{k-1}$，$k=2,3,\cdots,n$，则称序列是超递增序列，对应的背包称为简单背包。

超递增序列满足下列性质：

定理 5-4 设 $c_k = a_1x_1 + a_2x_2 + \cdots + a_kx_k$，则 $x_k = 1 \Leftrightarrow c_k \geqslant a_k$。

证明 若 $x_k = 1 \Rightarrow c_k = a_1x_1 + a_2x_2 + \cdots + a_{k-1}x_{k-1} + a_k \geqslant a_k$；

$x_k = 0 \Rightarrow c_k = a_1x_1 + a_2x_2 + \cdots + a_{k-1}x_{k-1} \leqslant a_1 + a_2 + \cdots + a_{k-1} < a_k$。定理证毕。

依据这一性质，我们可在 $O(n)$ 时间内解出简单背包问题。如下例所示。

【例 5-5】 求解背包问题：$2x_1 + 4x_2 + 9x_3 + 17x_4 + 36x_5 = 59$。

解：易验证 2，4，9，17，36 是一个超增序列。

$c_5 = 2x_1 + 4x_2 + 9x_3 + 17x_4 + 36x_5 = 59 > 36$， $\Rightarrow$ $x_5 = 1$，

$c_4 = 2x_1 + 4x_2 + 9x_3 + 17x_4 = 23 > 17$， $\Rightarrow$ $x_4 = 1$，

$c_3 = 2x_1 + 4x_2 + 9x_3 = 6 < 9$， $\Rightarrow$ $x_3 = 0$，

$c_2 = 2x_1 + 4x_2 = 6 > 4$， $\Rightarrow$ $x_2 = 1$，

$c_1 = 2x_1 = 2$， $\Rightarrow$ $x_1 = 1$，

最后得：$x_1 = 1$，$x_2 = 1$，$x_3 = 0$，$x_4 = 1$，$x_5 = 1$。

可见，简单背包问题是易解的，R.C.Merkl 和 M.E.Hellman 正是应用解一般背包问题的困难性、解简单背包问题的简单性、再加上适当的置乱技术来构造密码体制的。

5.5.3 背包密码体制的原理

设 $a_1,a_2,\cdots,a_n$ 是一个超递增序列，$b_1,b_2,\cdots,b_n$ 是一个普通序列，$a_i,b_i \in Z_n, i=1,2,\cdots,n$，并且，所有运算都在 Z_n 中进行。

令 $P = \{x_1a_1 + x_2a_2 + \cdots + x_na_n \mid x_i = 0,1, i=1,2,\cdots,n\} = < a_1,\cdots,a_n >$，

$C = \{x_1b_1 + x_2b_2 + \cdots + x_nb_n \mid x_i = 0,1, i=1,2,\cdots,n\} = < b_1,\cdots,b_n >$，

设 T：$P \to C$，T：$a_1x_1 + \cdots + a_nx_n \mapsto b_1x_1 + \cdots + b_nx_n$，满足：

$T(a_i)=b_i$，$i=1,2,\ldots,n$，

$T(a_1x_1+\ldots+a_nx_n)= T(a_1x_1+\ldots+a_nx_n)$。

是一个可逆置换。它的作用就是“隐藏” $a_1,a_2,\cdots,a_n$ 的超递增性，而改装其为 $b_1,b_2,\cdots,b_n$，T 可称为置乱变换。

将 $b_1,b_2,\cdots,b_n$ 公开。

对于给定的明文 $M = m_1m_2\cdots m_n$，$m_i = 0$ 或 1。

加密算法为：$c = b_1m_1 + b_2m_2 + \cdots + b_nm_n$。

对于密码分析者来说，要破译密文需要解一般背包问题：

$$b_1x_1+b_2x_2+\cdots+b_nx_n=c \text{。}$$

但对于合法信宿来说，作逆变换：$T^{-1}(b_1x_1+b_2x_2+\cdots+b_nx_n)=T^{-1}(c)$，得：

$a_1x_1+a_2x_2+\cdots+a_nx_n=T^{-1}(c)$。

这是一个简单背包问题，可在 $O(n)$ 时间内解出。

可见，置乱变换需要一种“线性性质”，且把“基” $a_1,a_2,\cdots,a_n$ 变换到“基” $b_1,b_2,\cdots,b_n$。乘积变换是一个合适的变换，应用这个变换就可得到 MH 背包密码体制。

5.5.4 MH 背包密码体制的描述

设 $(a_1,a_2,\cdots,a_n)$ 是一个超递增序列，取正整数 m，w，满足 $m>\sum_{k=1}^{n}a_k$，$(m, w)=1$。

令：$b_i\equiv wa_i \mod m$，$i=1,2,\cdots,n$。

定义线性变换 $T:<a_1,a_2,\cdots,a_n>\to<b_1,b_2,\cdots,b_n>$ 为：

$$T:\ x\mapsto wx \bmod m\text{。}$$

将超增序列 $(a_1,a_2,\cdots,a_n)$ 及整数 w 保密，而将序列 $(b_1,b_2,\cdots,b_n)$ 公开。

$b_1x_1+b_2x_2+\cdots+b_nx_n=c$ 是一个一般背包问题，分析者不能在多项式时间内求解这个问题。而合法信宿的解密过程如下：

在 Z_n 中作变换：

$w^{-1}(b_1x_1+b_2x_2+\cdots+b_nx_n)=w^{-1}c$，

$w^{-1}b_1x_1+w^{-1}b_2x_2+\cdots+w^{-1}b_nx_n=w^{-1}c$，

即：$a_1x_1+a_2x_2+\cdots+a_nx_n\equiv w^{-1}c \mod m$；

因为 $\left(w^{-1}c \mod m\right)<m$，$\sum_{k=1}^{n}a_k<m$，

所以：$a_1x_1+a_2x_2+\cdots+a_nx_n=\left(w^{-1}c \mod m\right)$。

这已是一个简单背包问题了，可容易求解。

我们将体制严格地描述如下：

设 $A=(a_1,a_2,\ldots,a_n)$ 是一个超递增序列，$p>\sum_{i=1}^{n}a_i$ 是一个素数，$1<w<p-1$，对 $1\leqslant i\leqslant n$，

定义：$b_i\equiv wa_i \bmod p$，且记：$B=(b_1,b_2,\ldots,b_n)$，设：

明文空间为：$P=\{0,1\}^n$，

密文空间：$C=\{0,1,2,\ldots,n(p-1)\}$，

密钥空间：$K=\{(A,p,w,B)\}$。B 公开，p，w，A 保密。

对 $k=(A,p,w,B)$，定义：

加密算法：$E_k(x_1,x_2,\cdots,x_n)=\sum_{i=1}^{n}b_ix_i$；

对 $0\leqslant y\leqslant p-1$，令 $z\equiv w^{-1}y \bmod p$，解简单背包问题：

$a_1x_1+a_2x_2+\ldots+a_nx_n=z$，定义：

解密算法：$d_k(y)=(x_1,x_2,\ldots,x_n)$。

在 80 年代初期，Merkle-Hellman 密码体制被 Shamir 使用 Lenstra 的整数规划算法破译了，为了加强体制的强度，Merkle 和 Hellman 提出，可以通过多次迭代的办法进一步隐藏超递增序列的特殊性质，而使破译问题变得更为复杂。方法如下：

设 $A=A^{(1)}=(a_k^{(1)})$ 是原始的背包序列（超递增序列），选择数对

(w_1,p_1)，(w_2,p_2)，…，(w_t,p_t)，

满足：p_i 为素数，$1\leqslant w_i<p_i-1$，$(w_i,p_i)=1$，$i=1,2,\ldots,t$，$p_1>\sum_{k=1}^{n}a_k^{(1)}$。

令：$A^{(i)}=\left(a_1^{(i)},a_2^{(i)},\cdots,a_n^{(i)},\right)$，$a_k^{(i)}\equiv w_{i-1}a_k^{(i-1)} \bmod p_{i-1}$，$k=1,2,\cdots,n$，$i=2,3,\ldots,t$。

最后定义

公开密钥为：

$$A=A^{(t)};$$

秘密密钥为：

$$(w_1^{-1},p_1),(w_2^{-1},p_2),\cdots,(w_t^{-1},p_t)，\quad A^{(1)}=(a_k^{(1)})。$$

用向量的内积来表示，则对于给定的明文 $M=(m_1,m_2,\ldots,m_n)$，可定义

加密变换为：

$$c=c^{(t)}=A^{(t)}M;$$

解密变换为：

（1）$A^{(i-1)}\equiv w_{i-1}^{-1}A^{(i)} \bmod p_{i-1}$，$\quad i=t,\ t-1,\cdots,2$，

（2）$c^{(i-1)}\equiv w_{i-1}^{-1}c^{(i)} \bmod p_{i-1}$，$\quad i=t,t-1,\cdots,2$，

（3）最后，解简单背包：$c^{(1)}=A^{(1)}M$，求得明文 M。

5.5.5 示例

设 $A=$（2,5,9,21,45,103,215,450,946）是一个秘密的超递增序列，$p=2003$，$w=1289$，则公开密钥是：

$B=$（575,436,1586,1030,1921,569,721,1183,1570）。

再设明文为：

$M=$（1,0,1,1,0,0,1,1,1）。

则密文为：

$c=(575+1586+1030+721+1183+1570) \bmod 2003=6665$。

解密过程为：

$z\equiv w^{-1}c \bmod p$

$\equiv 317\times 6665 \bmod 2003$

$\equiv 1643$

再解简单背包问题：

$2x_1+5x_2+9x_3+21x_4+45x_5+103x_6+215x_7+450x_8+946x_9=1643$，

解得明文：

$M=$（1,0,1,1,0,0,1,1,1）。

5.6 概率公钥密码体制

5.6.1 引言

一般地，公钥密码的一个明文对应于一个密文，因而密码分析者可以对密文进行选择明文攻击，即：可以选择任意明文，用公钥进行加密得到相应的密文，与原密文对照，如果相同，则相应的明文就是要求的明文。这种攻击对低熵明文空间非常有效。这种明文惟一确定密文的密码体制称为确定性密码体制。为了消除确定性密码体制的这种缺点，1982 年，美国加州大学伯克利分校的 S.Golwasser，S.Micali 提出了一种概率加密方法（简称为 GM 方法），即：同一明文被加密后可以得到不同的密文。这样，选择明文攻击就会失效。

科学发明、发现的历史说明，有时候提出想法比有了想法寻找实现的方法更重要。概率加密原理其实非常简单，用确定性加密算法也能轻易地实现概率加密算法。

设 E_k：$2^{128}\to 2^{128}$ 是确定性加密算法，简单地将明文空间分割为两个部分，一部分为明文空间，一部分为参数空间，如设 $2^{128}=2^{96}\times 2^{32}$，则得到加密算法：

$$E_k:\ 2^{96}\times 2^{32}\to 2^{128},$$

这就是一个明文为 96 比特的概率加密算法，其本质就是在明文后附上 32 比特随机信息，从而，同一个明文、密钥可以得到 2^{32} 个不同的密文。也容易看到，这种算法的代价是密文膨胀率的增大。从信息论的观点来看，这种膨胀是不可避免的。虽然 GM 方法不是基于确定性公钥算法的，但也不能避免这种膨胀性。

5.6.2 体制的描述

（1）体制的构造

信宿 B：任意选取两个强素数 p，q，令 $n=pq$；

任意选取一个正整数 $y\in Z_n^*$，满足：$\left(\dfrac{y}{p}\right)=\left(\dfrac{y}{q}\right)=-1$；

公布公钥：(y,n)，保密秘钥：(p,q)。

（2）加密算法

信源 A：获取 B 的公钥 (y,n)，将待发送信息 m 表示成二进制形式 $m=(m_1m_2\ldots m_s)$，$m_i\in\{0,1\},i=1,2,\ldots,s$；

对 $\forall m_i$，任选一个正整数 x_i，$1\leqslant x_i<n$，计算：

$$c_i=\begin{cases} yx_i^2 \bmod n\ , & m_i=1, \\ x_i^2 \bmod n\ , & m_i=0. \end{cases}$$

则密文为 $E(m)=(c_1,c_2,\ldots,c_s)$。

（3）解密算法

信宿 B：计算

$$m_i=\begin{cases}0, & \left(\dfrac{c_i}{p}\right)=\left(\dfrac{c_i}{q}\right)=1 \ ,\\ 1, & \left(\dfrac{c_i}{p}\right)=\left(\dfrac{c_i}{q}\right)=-1 \ .\end{cases}$$

解得明文：$m=(m_1m_2\ldots m_s)$，其中$\left(\dfrac{}{-}\right)$是 Jacobi 符号。

（4）解密原理：如果 m_i=1，则$\left(\dfrac{c_i}{p}\right)=\left(\dfrac{yx_i^2}{p}\right)=\left(\dfrac{y}{p}\right)=-1$，同理，$\left(\dfrac{c_i}{q}\right)=-1$；

如果 m_i=0，则$\left(\dfrac{c_i}{p}\right)=\left(\dfrac{x_i^2}{p}\right)=1$，同理，$\left(\dfrac{c_i}{q}\right)=1$。

（5）系统分析：攻击者不能分解 n，所以没有 p，q，也就不能解密。另一方面，由于$\left(\dfrac{c_i}{n}\right)=\left(\dfrac{c_i}{p}\right)\left(\dfrac{c_i}{q}\right)=1$，从而，由 Jacobi 符号值攻击者也不能得到任何明文比特信息。

由于明文的一个比特 m_i，被加密成 Z^*_n 中一个数，如果 n 是一个 155 位十进制正整数，一般地，n 有 512 比特，从而，密文的膨胀率达 512，显然这是不可忍受的。

Blum，S.Golwasser，Harn，Kiesler 等人提出了膨胀率较小的概率加密算法。

5.6.3 示例

（1）体制构造

1）设素数 $p=269$，$q=239$，$n=pq=64291=(1111101100100011)_2$，

易见，$|n|=|64291|=16$；

2）取 y=179，可验证 y 的 Jacobi 符号$\left(\dfrac{179}{269}\right)=\left(\dfrac{179}{239}\right)=-1$；

3）公钥为（179,64291），私钥为（269,239）。

设待加密信息为：Message= “gold”，则编码后为：

m=11001111101111110110011100100

（2）加密过程

$m_1=1, x_1=46482, c_1=179*46482^2 \bmod 64291=8586=(0001010)_2$，

$m_2=1, x_2=53128, c_2=179*53128^2 \bmod 64291=15983=(1101111)_2$，

$m_3=0, x_3=58413, c_3=58413^2 \bmod 64291=26617=(1111001)_2$，

$m_4=0, x_4=16257, c_4=16257^2 \bmod 64291=54039=(0010111)_2$，

$m_5=1, x_5=11487, c_5=179*11487^2 \bmod 64291=31671=(0110111)_2$，

$m_6=1, x_6=62248, c_6=179*62248^2 \bmod 64291=57551=(1001111)_2$，

$m_7=1, x_7=53185, c_7=179*53185^2 \bmod 64291=9770=(0101010)_2$，

$m_8=1, x_8=57327, c_8=179*57327^2 \bmod 64291=59418=(0011010)_2$，

$m_9=1, x_9=7779, c_9=179*7779^2 \bmod 64291=50859=(0101011)_2$，

$m_{10}=0, x_{10}=58867, c_{10}=58867^2 \bmod 64291=38789=(0000101)_2$，

$m_{11}=1, x_{11}=50679, c_{11}=179*50679^2 \mod 64291=43169=(0100001)_2$,

$m_{12}=1, x_{12}=32026, c_{12}=179*32026^2 \mod 64291=32743=(1100111)_2$,

$m_{13}=1, x_{13}=25416, c_{13}=179*25416^2 \mod 64291=13376=(1000000)_2$,

$m_{14}=1, x_{14}=45715, c_{14}=179*45715^2 \mod 64291=3691=(1101011)_2$,

$m_{15}=1, x_{15}=51294, c_{15}=179*51294^2 \mod 64291=17946=(0011010)_2$,

$m_{16}=1, x_{16}=46740, c_{16}=179*46740^2 \mod 64291=4466=(1110010)_2$,

$m_{17}=0, x_{17}=10166, c_{17}=10166^2 \mod 64291=31919=(0101111)_2$,

$m_{18}=1, x_{18}=39516, c_{18}=179*39516^2 \mod 64291=14552=(1011000)_2$,

$m_{19}=1, x_{19}=52741, c_{19}=179*52741^2 \mod 64291=19989=(0010101)_2$,

$m_{20}=0, x_{20}=49743, c_{20}=49743^2 \mod 64291=62623=(0011111)_2$,

$m_{21}=0, x_{21}=34250, c_{21}=34250^2 \mod 64291=8914=(1010010)_2$,

$m_{22}=1, x_{22}=49537, c_{22}=179*49537^2 \mod 64291=30285=(1001101)_2$,

$m_{23}=1, x_{23}=25819, c_{23}=179*25819^2 \mod 64291=55854=(0101110)_2$,

$m_{24}=0, x_{24}=30281, c_{24}=30281^2 \mod 64291=20719=(1101111)_2$,

$m_{25}=0, x_{25}=48068, c_{25}=48068^2 \mod 64291=42666=(0101010)_2$,

$m_{26}=1, x_{26}=44150, c_{26}=179*44150^2 \mod 64291=34495=(0111111)_2$,

$m_{27}=0, x_{27}=4384, c_{27}=4384^2 \mod 64291=60738=(1000010)_2$,

$m_{28}=0, x_{28}=46902, c_{28}=46902^2 \mod 64291=16748=(1101100)_2$,

于是得：

$E(m)=$ {8586, 15983, 26617, 54039, 31671, 57551, 9770, 59418, 50859, 38789, 43169, 32743, 13376, 3691, 17946, 4466, 31919, 14552, 19989, 62623, 8914, 30285, 55854, 20719, 42666, 34495, 60738, 16748}。

（3）解密过程

由 c_1=8586 得：

$$\left(\frac{c_1}{p}\right)=\left(\frac{8586}{269}\right)=\left(\frac{247}{269}\right)=\left(\frac{13\times19}{269}\right)=\left(\frac{13}{269}\right)\left(\frac{19}{269}\right)=-1,$$

$$\left(\frac{c_1}{q}\right)=\left(\frac{8586}{239}\right)=\left(\frac{221}{239}\right)=\left(\frac{13\times17}{239}\right)=\left(\frac{13}{239}\right)\left(\frac{17}{239}\right)=-1,$$

所以 $m_1=1$；

同理 c_2=15983 时，$\left(\frac{c_2}{p}\right)=\left(\frac{c_2}{q}\right)=-1$，所以，$m_2=1$；

再由 c_3=26617 得：

$$\left(\frac{26617}{269}\right)=\left(\frac{255}{269}\right)=\left(\frac{3}{269}\right)\left(\frac{5}{269}\right)\left(\frac{17}{269}\right)=1,$$

$$\left(\frac{26617}{239}\right)=\left(\frac{88}{239}\right)=\left(\frac{2}{239}\right)^3\left(\frac{11}{239}\right)=1,$$

所以 $m_3=0$。

同理可解得明文的其他比特。

5.7 NTRU 公钥密码体制

NTRU（Number Theory Research Unit）公钥密码体制是在 90 年代中期，由数学家小组 Jeffrey Hoffstein，Jill Pipher 和 Joseph Silverman 研制的。Hoffstein 博士首先在 CRYPTO'96 会议上提出并命名了 NTRU 加密算法，他印发了许多论文，在文章中详细地描述 Ntru 算法及初步的安全性分析，该文同时在因特网上发布。NTRU 算法、安全性分析的详细、完整的文本发表在“算术数论讨论会”文集中（the proceedings of the Algorithmic Number Theory Symposium (ANTS III, Portland 1998)）。

NTRU 被接受为 IEEE P1363 标准，被标准化在文档 Working Group for Standards In Public Key Cryptography 中。

NTRU 算法使用多项式代数及两个不同数的模。它的安全性基于多项式、不同模混合运算的相互作用，也依赖于最大格及寻找最短向量的困难性。

NTRU 加密、解密一个长度 N 的信息分组需要 $O\left(N^2\right)$ 次操作。RSA 需要 $O\left(N^3\right)$ 操作。所以，NTRU 比 RSA 快至少 100 倍。

5.7.1 NTRU 的工作空间

NTRU 密码体制依赖于三个参数 (N,p,q) 和 $N-1$ 次整系数多项式的 4 个集合 L_f, L_g, L_r, L_m，其中 p,q 不必是素数，但有 $(p,q)=1$，一般地假设 $q>p$。

（1）NTRU 密码体制工作空间为多项式环 $R = Z[X] \big/ \left(X^N-1\right)$。可见，

$$R=\left\{\sum_{i=0}^{N-1} F_i x^i \;\middle|\; F_i \in Z, i=0,1,\cdots,N-1\right\},$$

对任意 $F \in R$，可以记成向量形式 $F=\sum_{i=1}^{N-1} F_i x^i = \left[F_0, F_1, \cdots, F_{N-1}\right]$。

（2）对 $\forall f,g \in R$，$f=\left[f_0, f_1, \cdots, f_{N-1}\right]$，$g=\left[g_0, g_1, \cdots, g_{N-1}\right]$，

令：$h=f*g=\left(\sum_{i=0}^{N-1} f_i x^i\right)\left(\sum_{i=0}^{N-1} g_i x^i\right) \bmod x^N-1 = \sum_{i+j\equiv k \bmod N} f_i g_j x^k$

称为 R 上的星乘法。

记 $\left[h_0, h_1, \cdots, h_{N-1}\right]=\left[f_0, f_1, \cdots, f_{N-1}\right]*\left[g_0, g_1, \cdots, g_{N-1}\right]$，则星乘法可简单地写成矩阵形式：

$$\begin{pmatrix} h_0 \\ h_1 \\ h_2 \\ \vdots \\ h_{N-1} \end{pmatrix} = \begin{pmatrix} f_0 & f_{N-1} & f_{N-2} & \cdots & f_1 \\ f_1 & f_0 & f_{N-1} & \cdots & f_2 \\ f_2 & f_1 & f_0 & \cdots & f_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_{N-1} & f_{N-2} & f_{N-3} & \cdots & f_0 \end{pmatrix} \begin{pmatrix} g_0 \\ g_1 \\ g_2 \\ \vdots \\ g_{N-1} \end{pmatrix}$$

（3）对 $\forall f \in R$， $f = [f_0, f_1, \cdots, f_{N-1}]$，定义：

$f \bmod p = [f_0 \bmod p,\ f_1 \bmod p,\ \cdots, f_{N-1} \bmod p]$，称为多项式模运算。

5.7.2 密码体制

（1）密钥

随机地选择两个 $N-1$ 次多项式 $f, g \in L_g$。f 模 p, q 都有逆元。即存在 $F_q, F_p \in L_g$，满足：$F_p * f \equiv 1 \bmod p$，$F_q * f \equiv 1 \bmod q$。

令 $h \equiv F_q * g \bmod q$，则：

公钥就是 h；

私钥即为 f（F_p 也要保密）。

（2）加密算法

设信息为 $m \in L_m$，随机地选择一个多项式 $r \in L_r$，则：

加密算法：$c \equiv pr * h + m \bmod q$

（3）解密

首先计算 $a \equiv c * f \bmod q$，并选择 a 的系数在 $-\frac{q}{2}$，$\frac{q}{2}$ 之间。

其次计算 $m \equiv F_a * a \bmod p$。

5.7.3 解密原理与参数选择

（1）解密原理

$$\begin{aligned} f * c &\equiv f * pr * h + f * m \bmod q \\ &\equiv f * pr * F_q * g + f * m \bmod q \\ &\equiv pr * g + f * m \bmod q \end{aligned}$$

如果 f，g，r，m 的系数的绝对值都很小，则 $r*g$，$f*m$ 系数的绝对值相对于 q 也很小，又由于 $q>p$，所以，可以认为 $p\ r*g+f*m$ 的系数已在 $-\frac{q}{2}$，$\frac{q}{2}$ 之间，从而，如果选择 $a=(f*c \bmod q)$ 的系数也在 $-\frac{q}{2}$，$\frac{q}{2}$ 之间，则有：

$p\ r*g+f*m=(f*c \bmod q)$，从而：

$F_p*a \bmod p = p\ F_p*\ r*g+ F_p*f*m \bmod p= F_p*\ f*m \bmod p=m$。

于是得到明文 m。

但注意到：参数 f, g, r, m 的系数绝对值都必须很小，使 $pr*g+f*m$ 的系数在 $-\frac{q}{2}$，$\frac{q}{2}$ 之间，从而 $pr*g+f*m$ 模 q 不变，才能解密成功。否则，如果 f, g, r, m 的系数及 p, q 选择不当，则不能使 $pr*g+f*m$ 的系数在 $-\frac{q}{2}$，$\frac{q}{2}$ 之间，那么解密可能失败。

这时可用一个小的 x（x 可正可负），对 a 的系数作适当的修正，使之在 $-\frac{q}{2}+x$，$\frac{q}{2}+x$ 之间，再执行解密操作，但仍然有可能无法解密密文，不过这种概率很小。

（2）参数选择

首先定义 R 中元素 F 的宽度为：

$|F|_\infty = \max\limits_{1 \leqslant i \leqslant n}\{F_i\} - \min\limits_{1 \leqslant i \leqslant n}\{F_i\}$，这是一种 L_∞ 范数。

同理，定义 L^2 范数：$|F|_2 = \left(\sum\limits_{i=1}^{N}\left(F_i - \bar{F}\right)^2\right)^{1/2}$，$\bar{F} = \frac{1}{N}\sum\limits_{i=1}^{N} F_i$。

明文空间：$L_m = \left\{ m = \sum\limits_{i=1}^{N-1} m_i x^i \in R \,\middle|\, -\frac{1}{2}(p-1) \leqslant m_i \leqslant \frac{1}{2}(p-1) \right\}$

$L(d_1, d_2) = \left\{ F = \sum\limits_{i=0}^{N} F_i x^i \,\middle|\, d_1\text{个}\, F_i = 1,\ d_2\text{个}\, F_i = -1,\ \text{其余} F_i = 0 \right\}$。

选择三个正整数 d_f, d_g, d，令：$L_f = L\left(d_f, d_f - 1\right)$，$L_g = L\left(d_g, d_g\right)$，$L_r = L(d, d)$。

注意：不取 $L_f = L\left(d_f, d_f\right)$ 是因为 f 是可逆的，而满足 $f(1) = 0$ 的多项式不可逆。因为 $L_f = L\left(d_f, d_f - 1\right)$ 保证了 $f(1) \neq 0$。

对 $\forall f \in L_f, g \in L_g, r \in L_r$，它们的范数为：

$|f|_2 = \sqrt{2d_f - 1 - N^{-1}}$，$|g|_2 = \sqrt{2d_g}$，$|r|_2 = \sqrt{2d}$。

（3）解密条件

由解密原理可见，能够解密的条件是：$|pr * g + f * m|_\infty < q$。

还可以知道，如果 $|f * m|_\infty < \frac{q}{4}$，$|pr * g|_\infty < \frac{q}{4}$，则：$|pr * g + f * m|_\infty < q$。

定理 5-5 对任意 $\varepsilon > 0$，存在依赖于 ε，N 的常数 $\gamma_1, \gamma_2 > 0$，对 $\forall F, G \in R$，它们满足：$\gamma_1 |F|_2 |G|_2 \leqslant |F * G|_\infty \leqslant \gamma_2 |F|_2 |G|_2$ 的概率大于 $1 - \varepsilon$，即：

$$P\left(\gamma_1 |F|_2 |G|_2 \leqslant |F * G|_\infty \leqslant \gamma_2 |F|_2 |G|_2\right) > 1 - \varepsilon。$$

可见只要取 $|f|_2 |m|_2 \approx \frac{q\gamma}{4}$，$|r|_2 |g|_2 \approx \frac{q\gamma}{4}$，$\gamma$ 对应于一个小 ε 即可满足解密条件。

例如，实验表明，$N = 107$，$N = 167$，$N = 503$ 时，γ 对应的值为：0.35，0.27，0.17。

5.7.4 安全性分析

（1）穷举攻击

密钥穷举攻击：检验所有可能的 $f \in L_f$，并检验 $f * h \mod q$ 是否有小的分量，也可以检验所有可能的 $g \in L_g$，并检验 $g * h^{-1} \mod q$ 是否有小的分量。

秘密参数穷举攻击：检验所有可能的 $r \in L_r$，并检验 $e - r * h \mod q$ 是否有小的分量。

由于 $\#\left(L_g\right)$ 比 $\#\left(L_f\right)$ 小，所以，密钥空间容量为 $\#\left(L_g\right)$，秘密参数空间的容量为 $\#(L_r)$。然而，应用中间相遇攻击，则安全水平为：

密钥空间安全容量 $= \sqrt{\# L_g} = \frac{1}{d_g!}\sqrt{\frac{N!}{\left(N - 2d_g\right)!}}$，

秘密参数空间安全容量$=\sqrt{\#L_r}=\frac{1}{d\ !}\sqrt{\frac{N\ !}{(N-2d)\ !}}$

（2）对 NTRU 的其他攻击

对 NTRU 的攻击主要有：

Andrew Odlyzko 提出了对 r,f 的中间相遇攻击。

相同信息用不同密钥和秘密参数的多次传输攻击。

基于格的攻击。

（3）与 RSA 的比较

1）安全性：

RSA	NTRU	ECC
512	167	113
1024	263	160
2048	503	282

NTRU-167 安全年限 550 年，NTRU-503 安全年限 5.4×10^3 年。

2）速度：

NTRU 的密钥生成速度比 RSA 快 300 倍；

加密代码只须 20 行 C 代码，解密代码只须 52 行 C 代码；

NTRU 是所有其他公钥体制的 100 倍；

（4）NTRU 三种标准

实际的 NTRU 密码体制中有三种安全水平，在这三种安全级别中，f, g 的范数经过选择，使解密失败的概率小于 5×10^{-5}。

	N	q	p
中等安全	167	128	3
标准安全	251	128	3
高级安全	347	128	3
最高安全	503	256	3

1）中等安全级别参数

这一安全水平适合于较短的秘密信息，密钥常常可适当地变化。可用于寻呼机、移动电话传输等。

这些参数为：

$$(N,p,q)=(107,3,64),\quad L_f=L(15,14),\quad L_g=L(12,12),\quad L_r=L(5,5)。$$

密钥的比特长度：秘钥=340 比特，公钥=642 比特。

（抗相遇攻击的）安全水平为：密钥空间容量$=2^{50}$，信息空间容量$=2^{26.5}$。

2）高级安全水平

这些参数为：

$$(N,p,q)=(167,3,128),\quad L_f=L(61,60),\quad L_g=L(20,20),\quad L_r=L(18,18)。$$

密钥的比特长度：秘钥=530 比特，公钥=1169 比特。

（抗相遇攻击的）安全水平为：密钥空间容量$=2^{82.9}$，信息空间容量$=2^{77.5}$。

3）最高安全水平

$(N,p,q)=(503,3,256)$，$L_f=L(216,215)$，$L_g=L(72,72)$，$L_r=L(55,55)$。

密钥的比特长度：私钥=1595 比特，公钥=4024 比特。

（抗相遇攻击的）安全水平为：密钥空间容量=2^{285}，信息空间容量=2^{170}。

5.7.5 示例

取 $N=11$，$q=32$，$p=2$，$d_f=4$，$d_g=3$，$d_r=3$。

即：$L_f=L(4,3)$，$L_g=L(4,4)$，$L_r=L(3,3)$。

令：$f=-1+x+x^2-x^4+x^6+x^9-x^{10}$，$g=-1+x^2+x^3+x^5-x^8-x^{10}$，

$F_p=1+2x+2x^3+2x^4+x^5+2x^7+x^8+2x^9$，

$F_q=5+9x+6x^2+16x^3+4x^4+15x^5+16x^6+22x^7+20x^8+18x^9+30x^{10}$。

易验证：$F_p*f\equiv 1 \bmod p$，$F_q*f\equiv 1 \bmod p$。

公钥为：

$h=pf_q*g \bmod 32$

$=8+25x+22x^2+20x^3+12x^4+24x^5+15x^6+19x^7+12x^8+19x^9+16x^{10} \bmod 32$。

设需要传输的信息为：$m=-1+x^3-x^4-x^8+x^9+x^{10}$。

（可以将 m 表示成二进制后，每一位再减去 1 得到 m 在$\{-1，0，1\}^{10}$上的编码。）

随机地选取：$r=-1+x^2+x^3+x^4-x^5-x^7$。

则密文为：$c=r*h \bmod 32$

$=14+11x+26x^2+24x^3+14x^4+16x^5+30x^6+7x^7+25x^8+6x^9+19x^{10} \bmod 32$。

解密：$a=f*c \bmod 32$

$=3-7x-10x^2-11x^3+10x^4+7x^5+6x^6+7x^7+5x^8-3x^9-7x^{10} \bmod 32$。

注意到，选取 a 的系数在-15，16之间。

$b=a \bmod p$

$=-x-x^2+x^3+x^4+x^5+x^7-x^8-x^{10} \bmod 3$。

最后得到解密的明文为：

$m=F_p*b \bmod p$

$=-1+x^3-x^4-x^8+x^9+x^{10} \bmod 3$。

这就完成了解密。

5.8 公钥密码标准

1977 年，Ron Rivest、Adi Shamir 和 Len Adleman 首次在《科学美国人》杂志上公开发表了他们的 RSA 公钥密码体制以后，RSA 得到了广泛的应用。他们申请了专利，开设了自己的公司，称为 RSA 数据安全公司。这也是第一家研发公钥密码产品的公司，并成为公钥密码产品行业的领头企业，在密码学界产生了极大的影响。从 1991 年起，他们相继推出了一系列公开密钥密码标准（PKCS，Public-Key Cryptography Standards），被引为业界标准，如 IETF，ANSI 标准都兼容了 PKCS，值得一提的是，由于美国国家安全局（NBS）一直没有介入公钥密码的标准化工作，所以 PKCS 已经成为事实上的（美国）国家标准和国际标准。

PKCS 目前共有 15 个标准，其中 PKCS#13 是关于椭圆曲线公钥标准，但正式标准方案还没有公布。下面，我们就 PKCS#1，PKCS#3 部分主要内容进行描述。

5.8.1 PKCS #1

PKCS #1 v2.1 是 RSA 公钥密码标准（RSA Cryptography Standard），该版本由 RSA 公司于 2002 年 6 月 14 日颁布，旨在代替 PKCS #1 V2.0 但与 PKCS #1 V2.0 兼容。该标准也与标准 IEEE-1363-2000 和 ANSI X9F1 标准草案兼容。该标准描述低层 RSA 基本的加密、签名方式，主要与其他安全协议一起应用，定义下层 RSA 运算。

PKCS#1V2.1 支持所谓的多素数模的 RSA，也就是说，RSA 模数可以有多个因子。但是，操作上它只对密钥的生成有影响。这种 RSA 称为多重素数 RSA。该体制的优点是应用孙子定理减少了解密、签名的计算量。

该标准分几个部分，下面分别介绍。

1．PKCS#1 定义的公钥数据格式

```
RSAPublicKey::=SEQUENCE{
        modulus           integer    // 模数 n
        publicExponent    integer    // 公钥 e
}
```

2．PKCS#1 定义的私钥数据格式

```
RSAPrivateKey::=SEQUENCE{
        version           integer    版本值，取 0
        modulus           integer    // 模数 n
        PublicExponent    integer    // 公钥 e
        privateExponent   integer    // 私钥 d
        prime1            integer    // 素数 p
        prime2            integer    // 素数 q
        exponent1         integer    // d 模 p-1
        exponent1         integer    // d 模 q-1
        coefficient       integer    // q^-1 模 p
}
```

3．I2OSP

I2OSP 将一个非负整数转化成一个指定长度的字符串：I2OSP ($x, xLen$)；

输入：x——要被转化的非负整数；

$xLen$——指定的输出字符串的长度；

输出：X——具有指定长度 $xLen$ 的输出字符串；

转化步骤如下：

1）如果 $x \geq 256^{xLen}$，输出 “integer too large”，停止；

2）将 x 表示成 256-进制，具有长度 $xLen$，

$x = x_{xLen-1}256^{xLen-1}+x_{xLen-2}256^{xLen-2}+\ldots+x_1 256+x_0$，$0 \leq x_i < 256$；

3）输出：$X = X_1X_2\ldots X_{xLen}$，$1 \leq i \leq xLen$，即将 x 表示成字符串；

4. OS2IP

OS2IP 将字符串转化成一个非负整数：OS2IP (*X*)；

输入：*X*——要被转化的字符串；

输出：*x*——转化后的非负整数；

转化步骤如下：

1）对输入 $X_1X_2\ldots X_{xLen}$ 计算：$x = x_{xLen-1}256^{xLen-1}+x_{xLen-2}256^{xLen-2}+\ldots+x_1 256+x_0$；

2）输出 x。

5. 密码原算子

加、解密原算子是指加、解密的基本数学计算，记作 RSAEP/RSADP，但 RSAEP 和 RSADP 所表示的数学运算是一样的，仅仅是输入上有差别。

同样，数字签名与验证的原算子也是指数字签名与验证的基本算子。记作 RSASP1/RSAVP1。本质上，IFSP-RSA1/IFVP-RSA1 和 RSAEP/RSADP 一样。

（1）RSA 加密算子 RSAEP：RSAEP ((*n*, *e*), *m*)

输入：(*n*, *e*) ——接收端的 RSA 公钥；

m——被加密的信息，$0 \leqslant m \leqslant n-1$；

输出：c——加密后的密文，$0 \leqslant c \leqslant n-1$；

假定：RSA 公钥是有效的。

步骤如下：

1）如果 m 不在 0，n–1 之间，则输出错误信息"message representative out of range"并停止；

2）计算：$c = m^e \bmod n$;

输出：c。

（2）RSA 解密算子 RSADP：RSADP (*K*, *c*)

输入：*K*——RSA 私钥，可以是以下三种之一：

1）(*n*, *d*)，

2）(*p*, *q*, *dP*, *dQ*, *qInv*)，

3）(r_i, d_i, t_i), i = 3, …, u。

c——加密后的密文，$0 \leq c \leq n-1$；

输出：m——解密后的信息，$0 \leq m \leq n-1$；

假定：RSA 私钥 *K* 是有效的。

步骤如下：

1）如果密文 c 不在 0 和 n–1 之间，则输出 "ciphertext representative out of range" 且停止；

2）对第一种密钥用 (*n*, *d*)，计算 $m = c^d \bmod n$;

对于第二种密钥 (*p*, *q*, *dP*, *dQ*, *qInv*)，计算 $m_1 = c^{dP} \bmod p$，$m_2 = c^{dQ} \bmod q$，$h = (m_1 - m_2) \cdot qInv \bmod p$，$m = m_2 + q \cdot h$。

对于第三种密钥 (r_i, d_i, t_i)，计算 $m_i = c^{d_i} \bmod r_i$, $i = 3, \ldots, u$。

对 i =3 到 u，令 $R = r_1$，$R = R \cdot r_{i-1}.$，$h = (m_i - m) \cdot t_i \pmod{r_i}$，$m = m + R \cdot h.$；

3）输出 m。

（3）RSA 数字签名算子 RSASP1：RSASP1 (K, m)

输入：K——RSA 私钥，可以是以下形式之一：

第一种形式 (n, d)，

第二种形式 $(p, q, dP, dQ, qInv)$，

第三种形式(r_i, d_i, t_i), $i = 3, \ldots, u$,

m——需要被签名的信息，在 0，n–1 之间的非负整数。

输出：s——数字签名后的信息，在 0，n–1 之间的非负整数。

假定：RSA 私钥 K 是有效的。

步骤如下：

1）如果信息 m 在 0，n–1 之间，则输出“message representative out of range”，且停止；

2）数字签名被计算如下：

对于第一种密钥 (n, d) ，计算 $s = m^d \bmod n$，

对于第二种密钥$(p,q,dP,dQ,qInv)$，计算 $s_1 = m^{dP} \bmod p$ 和 $s_2 = m^{dQ} \bmod q$，$h = (s_1 - s_2) \cdot qInv \bmod p$，$s = s_2 + q \cdot h$.；

对于第三种密钥(r_i, d_i, t_i)，计算：$s_i = m^{d_i} \bmod r_i$, $i = 3, \ldots, u$，令 $R = r_1$ ，对 $i = 3$ 到 u ，作：$R = R \cdot r_{i-1}$.，$h = (s_i - s) \cdot t_i \pmod{r_i}$，$s = s + R \cdot h$；

3）输出：s。

（4）RSA 数字签名验证算子 RSAVP1：RSAVP1 $((n, e), s)$

输入：(n, e) ——接收端 RSA 公钥；

s——签名信息，在 0，n–1 之间；

输出：m——输出信息，在 0，n–1 之间；

假定：RSA 公钥(n, e)是有效的。

步骤如下：

1）如果签名信息 s 在 0，n–1 之间，则输出 “signature representative out of range”，且停止；

2）计算：$m = s^e \bmod n$；

3）输出：m。

6. RSA 编码加密过程 RSAES-OAEP

RSAES-OAEP 结合了加、解密原算子 RSAEP/RSADP 和编码方法 EME-OAEP，与定义在 IEEE Std 1363-2000 种的 IFES 兼容。IFES 是加、解密原算子 IFEP-RSA /IFDP-RSA 与编码方法 EME-OAEP 的结合。RSAES-OAEP 以 $k - 2hLen - 2$ 字节作输入， $hLen$ 是散列函数输出的长度， k 是 RSA 模数的字节数。

RSA 加密程序：RSAES-OAEP-Encrypt $((n, e), M, L)$

选项：Hash 散列函数；

MGF 马赛克生成函数；

输入：(n, e)——RSA 公钥，k 为 n 的字节数；

M——要被被加密的信息，长度为 mLen，$\text{mLen} \le \text{k} - 2\text{hLen} - 2$；

L——伴随于信息的标志，如果 L 没有提供，则 L 的默认值为空字符串。

输出：C——密文，长度为 k 的字符串。

假定：RSA 公钥 (n, e) 是有效的。

步骤如下：

（1）长度检验

如果 L 的长度比散列函数的输入限制大（例如 SHA-1 的限制是 $2^{61}-1$），则输出“label too long”，停止；

如果 *mLen>k–2hLen*–2,，则输出“message too long”，并停止；

（2）EME-OAEP 编码

1）如果标志 *L* 没有给定，则 *L* = “ ”，*lHash* = Hash (*L*)是一个长度 *hLen* 的字符串；

2）生成一个长度 *k–mLen–2hLen*–2 的 0 字符串 *PS*，*PS* 的长度能是 0。

3）连接 *lHash*，*PS* 和十六进制数 0x01，信息 *M*，得到一个长度 *k–hLen*–1 的字符串：*DB* = *lHash* || *PS* || 0x01 || *M*。

4）生成一个长度 *hLen* 的种子 *seed*；

5）*dbMask* = MGF (*seed*, *k–hLen*–1)；

6）maskedDB = DB ⊕ dbMask；

7）seedMask = MGF (maskedDB, hLen)；

8）maskedSeed = seed ⊕ seedMask；

9）连接十六进制 0x00，*maskedSeed* 和 *maskedDB* 形成长度 *k* 的字符串：*EM* = 0x00 || *maskedSeed* || *maskedDB*；

编码算法如图 5-1 所示。

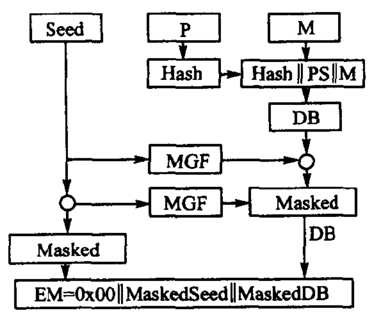


图 5-1　EME-OAEP 编码算法

10）如果 *L* = “ ”，则 *lHash* = Hash (*L*)对不同的散列函数分别有值：

SHA-1：	(0x)da39a3ee	5e6b4b0d	3255bfef	95601890	afd80709；
SHA-256：	(0x)e3b0c442	98fc1c14	9afbf4c8	996fb924	27ae41e4
	649b934c	a495991b	7852b855		
SHA-384：	(0x)38b060a7	51ac9638	4cd9327e	b1b1e36a	21fdb711
	14be0743	4c0cc7bf	63f6e1da		

（3）RSA 加密

1）转化 EM 成为正整数 *m*：*m* = OS2IP (EM) ；

2）应用 RSAEP 到公钥 (n, e) 和信息 m 产生密文 c：c = RSAEP ($(n, e), m$)；

3）转化密文 c 成长度为 k 的字符串 C ：C = I2OSP (c, k) ；

4）输出密文：C。

RSA 解密程序：RSAES-OAEP-Decrypt (K, C, L)

选项：Hash 散列函数，输出为长度 *hLen* 的字符串；

MGF 马赛克生成函数；

输入：K——接收端的 RSA 私钥，k 表示 RSA 模数 n 的字节数；

C——需要解密的，长度为 k 的密文，$k \geq 2hLen+2$；

L——可选的，伴随于信息的检验标志，默认值为空字符串；

输出：M——解密后的信息，字节数为 *mLen*，$mLen \leq k-2hLen-2$；

步骤如下：

（1）长度检验

如果 L 的长度比散列函数的输入限制大（例如 SHA-1 的限制是 $2^{61}-1$），则输出“decryption error”，停止；

密文 C 不是 k 个字节，则输出“decryption error”，停止；

如果 $k<2hLen+2$,，则输出“decryption error”，并停止；

（2）RSA 解密

1）将密文字符串 C 转化为正整数 c：c = OS2IP (C)；

2）应用 RSADP 到 RSA 私钥 K 和密文 c 恢复信息 m：m = RSADP (K, c)，如果 RSADP 输出 “ciphertext representative out of range” (即：$c \geq n$),，则输出 “decryption error”，停止；

3）转化信息 m 成为 k 字节的编码信息 *EM*：*EM* = I2OSP (m, k) ；

4）EME-OAEP 解码：分割编码信息 *EM* 成为字节 Y, *hLen* 个字符的字符串 *maskedSeed* 和长度为 $k-hLen-1$ 的字符串 *maskedDB* ：*EM* = Y ‖ *maskedSeed* ‖ *maskedDB*；

5）计算：seedMask = MGF (maskedDB, hLen)；

6）计算：seed = maskedSeed ⊕ seedMask；

7）计算：*dbMask* = MGF (*seed*, $k-hLen-1$)；

8）计算：DB = maskedDB ⊕ dbMask；

9）分割 *DB* 成为一个长度 *hLen* 的字符串 *lHash'*，一个取值 0x00 的填充串 *PS* （可能是空值），及信息 M：*DB* = *lHash'* ‖ *PS* ‖ 0x01 ‖ M；

10）如果在填充串 *PS* 和信息 M 之间没有十六进制 0x01 分割标志，或者 *lHash* 不等于 *lHash'*，或 Y 是非 0，则输出“decryption error”，并停止；

11）输出信息 M。

5.8.2 PKCS#3 v2.1

PKCS#3 描述 Diffie-Hellman 密钥交换标准，该标准用于网络上进行安全密钥交换，可以实现网络上两个不认识对象之间的保密通信。

PKCS#3 的参数（大写表示字符，小写表示整数）如表 5-2 所示：

表 5-2 PKCS#3 参数表

PV 公钥	p 素数
PV' 辅助公钥	x 私钥
SK 私钥	x' 辅助私钥
g 基低	y 整数公钥
k 素数 p 长度字节数	y' 其他整数公钥
l 私钥 x 长度比特数	z 整数私钥
n 模	

其中 $0<g<p$，且 $2^{8(k-1)}<p<2^{8k}$，$2^{l-1}<x<2^{l}$。

PKCS#3 的公钥数据格式如下：

```
DHParameter::=SEQUENCE{
        prime                   integer     // 素数p
        base                    integer     // 基低g
        privateValueLength      integer     //可选私钥长度
    }
```

Diffie-Hellman 协议分成两个阶段，对于每个阶段，通信的双方可以独立执行：

（1）用户产生信息 PV，并向对方发送。

1）双方各自选取秘密值 x，$2^{l-1}<x<2^{l}$；

2）计算 $y=\left(g^{x} \bmod p\right)$；

3）将整数转化成字符串 $y=\sum_{i=1}^{k} 2^{8(k-1)} PV_i$，即令 $PV=\left(PV_1, PV_2, \cdots, PV_k\right)$；

4）将 PV 传送给用户 B。

（2）接收对方信息 PV'，并生成公享私钥 SK。

1）将 PV'转化成整数：$y'=\sum_{i=1}^{k} 2^{8(k-1)} PV_i'$；

2）计算 $z=\left(y'^{x} \bmod p\right)$；

3）将 z 转化成 k 字节的私钥 SK，$z=\sum_{i=1}^{k} 2^{8(k-1)} SK_i$；

4）公享私钥为：$SK=\left(SK_1, SK_2, \cdots, SK_k\right)$。

5.9 习题

1. 设 $M_{p,3\times3}=\left\{\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \middle| a_{ij} \in Z_{17}, i, j=1,2,3\right\}$ 是 Z_{17} 上 3 阶非奇异方阵所成的群，在 $M_{p,3\times3}$ 上实现 Diffie-Hellman 密钥交换协议。

2. 在 Z^*_{97} 上实现 Diffie-Hellman 密钥交换协议。

3．研究求强素数计算的时间、空间复杂性。

4．设 p，q 是两个不同的素数，$n = pq$，$p-1 | ed-1$，$q-1 | ed-1$，证明对任意 x，$x^{ed} \equiv x \mod n$。

5．对于任意的素数 p，q，$n = pq$，证明存在坏密钥 e，对任意 x，$x^e \equiv x \mod n$。

6．设 p=73，q=37，n=2701，e=3 实现 RSA 公钥密码体制，并加密信息 m=1234。

7．设 p=7853，q=8753，n=68737309，e=3 实现 RSA 公钥密码体制，并加密信息 m=12345678。

8．设 p=37853，q=28753，n=1088387309，e=3 实现 RSA 公钥密码体制，并加密信息 m=12345678。

9．用 p-1 方法分解因子：

（1）n=766954236977107042O3；

（2）n=1028882337099941147485491092O535703；

（3）n=71364019841694637293908584452061371737。

10．用素因子之差法分解整数：

（1）n=1394798309；

（2）n=2338689071。

11．设 p=179，q=263，n= pq=47077，用 Rabin-47077 密码体制，加密信息 m=12345。

12．设 p=2347，q= 2551，n= pq=5987197，用 Rabin-5987197 密码体制，加密信息 m=123456。

13．设 p=239，q=607，n=pq= 145073，用 Rabin-145073 密码体制，解密信息 m=142271。

14．设 p=3187，q= 2551，n=pq= 22815733，用 Rabin-22815733 密码体制，解密信息 m=21557029。

15．设 p=541，g=357 是 Z^*_p 的生成元，构造 ElGamal-541 密码体制；

16．设 p=2063，g=877 是 Z^*_p 的生成元，取秘密密钥为 a=37，随机数 k=747，构造 ElGamal-2063 密码体制，并加密信息 m=860。

17．设 p=7309，g=877 是 Z^*_p 的生成元，取秘密密钥为 a=37，公开密钥 b=6689，构造 ElGamal-7309 密码体制，解密密文 c=(3505,4903)。

18．设 p=127373，g=877 是 Z^*_p 的生成元，取秘密密钥为 a=711，公开密钥 b= 77286，构造 ElGamal-127373 密码体制，解密密文 c=(47805,21057)。

19．解简单背包问题：$17x_1+222x_2+3878x_3+49212x_4+12783x_5+31728x_6+53342x_7+71277x_8=134521$。

20．用简单背包 (17,237,3878,49212,12783,31728,53342,71277) 构造一个 MH-背包密码体制。

21．设 A={19,21,49,157,451,10011,71234,765434}为 MH 背包秘钥，m=776543，w=7827，B={148713,164367,383523,452296,423805,701797,767187,22673}为 MH 背包公钥，解密密文 c=1956368。

22．设 Golwasser-Micali 概率密码体制的公钥为：n=pq= 32754889，y=54，加密信息 m=555。

23．设 Golwasser-Micali 概率密码体制的公钥为：n=pq= 2263443023，y=766，加密信

息 m=23。

24．设 Golwasser-Micali 概率密码体制的秘钥为：p=503，q=383，公钥为：$n=pq$= 192649，y=53，解密信息 m={178077, 113842, 147938, 98536, 11229, 115141, 109101, 75877, 123975}和 m={9684, 93133, 124219, 75459, 115503, 6533, 141153, 47932, 50261, 78026}。

25．构造一个 NTRU(N,p,q)=NTRU(13,3,33)。

26．设 N=13，p=3，q=32，已知 NTRU（13,3,32）的秘密密钥为：f=[1,−1,1,0,1,−1,1,−1, −1, 0,1,0,0]，F_p=[2,1,1,1,1,0,0,2,2,2,2,2,0]，F_q=[11,0,26,13,6,19,13,15,7,19,8,19,5]，g=[1,−1,1,0,1,0,1, −1,−1,0,−1,0,0]，经计算得公钥：

h=[10,21,6,11,21,23,14,30,21,20,23,7,17]，加密以下消息，并进行解密验证：

（1）随机元为：r=[1,−1,1,0,1,0,1,−1,−1,0,−1,0,0]，消息为：m=[1,1,0,−1,1,0,1,−1,0,0,1,1,1]；

（2）随机元为：r=[−1,−1,0,1,0,−1,0,−1,0,0,1,1,1]，消息为：m=[1,1,−1,0,0,1,−1,−1,0,−1,−1,−1,−1]。

27．设 N=11，p=3，q=32，已知 NTRU（11,3,32）的秘密密钥为：

f=[0,−1,1,1,−1,−1,0,1,1,0,0]，F_p=[2,2,1,2,1,2,1,1,2,0,2]，F_q=[4,22,12,18,3,20,29,22,30,1,0]，参数 g=[0,−1,1,1,0,−1,0,1,−1,0,0]。计算公钥：h，加密以下消息，并进行解密验证：

（1）随机元为：r=[0,−1,1,1,0,−1,0,1,−1,0,0]，消息为：m=[0,1,1,1,1,−1,0,1,−1,0,1]；

（2）随机元为：r=[0,−1,0,0,1,1,0,0,−1,−1,1]，消息为：m=[1,1,0,−1,1,1,1,0,−1,−1,0]；

（3）随机元为：r=[0,0,−1,1,0,0,−1,−1,1,1,0]，消息为：m=[−1,1,−1,0,1,−1,0,1,−1,−1,0]；

28．设 N=23，p=3，q=64，考虑 NTRU（23,3,64），

（1）已知 f=[0,−1,1,−1,0,−1,−1,0,0,1,0,0,−1,−1,1,0,1,1,1,1,−1,1,0]，求：F_p，F_q；

（2）已知参数 g=[0,−1,−1,−1,0,0,−1,0,0,1,0,0,−1,−1,1,0,1,1,1,1,−1,1,0]，求公钥：h；

（3）已知随机元为：r=[−1,1,0,1,0,−1,−1,0,1,−1,−1,1,0,−1,−1,1,0,1,0,1,0,0,0]，加密消息：m=[1,0,−1,−1,0,−1,−1,−1,1,0,1,0,1,−1,1,1,−1,−1,−1,−1,−1,−1,0]；

（4）已知随机元为：r=[1,0,−1,1,0,0,−1,0,0,0,1,1,−1,0,1,0,0,−1,1,−1,1,−1,−1]，加密消息：m=[1,0,0,1,−1,1,0,1,1,1,1,−1,−1,1,1,0,−1,1,−1,0,0,0,1]

29．设 N=29，p=3，q=64，考虑 NTRU（29,3,64），已知秘密密钥为：f=[−1,−1,0,0,1,1,−1, 0,0,−1,0,0,−1,0,0, −1,1,1,−1,1,1,1,1,−1,0,0,1,0,0]，F_p=[0,2,2,2,2,1,2,1,1,0,1,1,2,1,0,1,2,2,1,2,0,0,2, 1,2,0,0,0,0]，

（1）解密消息：

c_1=[46,7,44,38,12,24,15,42,2,55,42,11,39,12,17,47,2,44,7,10,9,1,33,24,11,32,7,26,43]；

（2）解密消息：

c_2=[46,6,45,37,14,24,16,42,4,55,41,10,39,13,17,49,1,44,6,10,9,0,32,23,12,31,7,27,43]；

（3）解密消息：

c_3=[45,7,45,38,14,22,15,44,4,56,42,12,37,11,17,48,2,45,6,9,9,1,34,23,10,33,9,28,43]；

（4）解密消息：

c_4=[46,6,44,38,13,23,15,43,3,55,42,11,38,12,17,49,2,46,7,11,11,2,34,24,12,33,9,28,44]。

5.10 实验题

1．编写一个可行的求强素数的计算机程序，求取 80 位十进制的强素数。

2. 完整实现 RSA 公钥密码体制。
3. 完整实现 Rabin 公钥密码体制。
4. 完整实现 ElGamal 公钥密码体制。
5. 完整实现 NTRU(N,p,q)=NTRU(167,3,128)公钥密码体制。

第 6 章　数字签名方案与散列函数

6.1　数字签名方案

6.1.1　引言

我们每天都在使用签名，如签订合同，在银行取款，批复文件等等。但这些都是手写签名。数字签名是电子信息技术的产物，是相应于电子文件签名的一种方法，是电子文件，数字对象的合法性、真实性标记和签名者的承诺。数字签名被广泛地应用于电子商务，（网络）身份认证，甚至核试验条约的检验等，已成为人们越来越不可缺少的活动。

首先我们来分析一下传统文件、电子文件及其签名的差别。

任何一个文件都有信息、符号、物理载体等几个要素。信息是文件的内涵，表示文件携带的意义或内容；符号是文件记录的基本形式，传统文件的符号一般都是墨点图形，具有物理特征，电子文件的符号是二进制编码，具有逻辑形式；传统的文件的物理载体一般都是纸质的，而电子文件的物理载体具有电磁形式。这些特征，决定了对它们进行签名的方法的不同。

传统纸质文件签名中，签名与文件使用的都是物理符号，且具有共同的物理载体。物理符号、载体的不可分割，惟一性（不可复制）保证了签名与文件的不可分割（伪造）、重复使用。

但是，在电子文件签名中，符号是逻辑的 0，1 串，物理载体具有电磁形式，所以没有通常的物理形状。而电磁载体是可以任意分割、复制的，所以，如果依照传统方式制作电子文件签名，那么，签名与文件可以分割，从而可以任意修改文件内容；文件也可以任意复制，从而可以被重复使用，完全达不到签名的目的。

另一方面，传统签名符号（一般为手写符号）的验证是通过与存档手迹对照来确定真伪的。它是主观的、模糊的，从而是容易伪造的，不安全的。

电子文件是一个二进制编码序列，对它的签名也只能是一种二进制编码序列，所以，对它的验证应该是客观的、精确的。现实中，我们采用加密、散列函数等技术对文件实施签名，我们将这种电子文件的签名方法称为数字签名。

由于物理性质的差别，数字签名应该具有如下性质：

（1）签名与文件应该是一个不可分割的整体，这样可以防止签名被分割后替换文件、替换签名等形式的伪造；

（2）数字签名应该具有时间特征，防止签名文件的重复使用；

（3）对于一个文件，签名应可以精确、惟一地生成，公开、精确、有效地验证。

一个签名方案由两部分组成：带有陷门的公开签名算法和验证算法。

本质上讲，数字签名算法是一个由密钥控制的函数。对任意一个消息 x，一个密钥 k，

签名算法产生一个签名 $y=sig_k(x)$，算法是公开的，但密钥是保密的，这样，不知道密钥的人不可能产生正确的签名，从而不能伪造签名。验证算法 ver 也是公开的，它通过 $ver(x, y)$=true 或 false 来验证签名。签名算法与验证算法都应该是多项式时间函数。

签名方案的定义是：

定义 6-1 设 P 是消息的有限集合，A 是签名的有限集合，K 是密钥的有限集合，则：

签名算法是一个映射：

$$sig: P\times K\to A,\quad sig:(x,k)\mapsto sig_k(x),$$

验证算法也是一个映射：

$$ver: P\times A\to\{\text{True},\ \text{False}\},\quad ver(x,y)=\begin{cases}\text{true}, & y=sig_k(x)\\ \text{false}, & y\neq sig_k(x)\end{cases}$$

五元组 $\{P,A,K,sig,ver\}$ 就称为一个签名方案。

签名算法一般由加密算法来充当。与其他分组形式密码体制一样，签名方案也只是计算上安全的，因为对于给定的消息 x，只须对所有 x 的可能签名 y，利用公开验证算法 ver 测试 $ver(x, y)$，直到 $ver(x, y)$=true，则可确定该 y 就是 x 的签名，这个 y 就是伪造的签名。

最基本的数字签名有基于对称密钥密码算法和基于公开密钥密码算法两种。

（1）基于对称密钥密码算法的数字签名方法

基于私钥密码算法的数字签名方法的本质是共享密钥的验证。

基本形式是：用户 A 与 B 共享对称密钥密码体制的密钥 k，要签名的消息为 m。则签名算法就是加密算法：

$$y=sig_k(m)=E_k(m)\text{。}$$

验证算法就是解密算法：

$$Ver(m,y)=True\Leftrightarrow m=D_k(y),$$

或加密算法：

$$Ver(m,y)=True\Leftrightarrow y=E_k(y)\text{。}$$

因为只有 A 才有密钥 k，所以，验证原理是显然的。

（2）基于公钥密码算法的数字签名方法

基于公钥密码算法的数字签名算法本质上是公钥密码算法的颠倒应用。

用户 A 选定公钥密码体制：加密密钥 e，解密密钥 d，加、解密算法 E_e、D_d，将 e 保密，称为签名密钥；将 d 公开，称为验证密钥，则签名和验证过程为：

设用户 A 要向用户 B 发送消息 m，并签名：

1）用户 A 用加密密钥加密（签名）消息 m，$y=E_e(m)$，向 B 发送 (m,y)；

2）用户 B 用 A 的公钥（解密密钥）验证（解密）签名：

$$Ver(m,y)=True\Leftrightarrow m=D_d(y)\text{。}$$

原理是显然的。

6.1.2 RSA 签名方案

RSA 公钥体制是一个既可应用于加密，也可应用于签名的密码方案。

RSA 签名方案定义如下：

设 p,q 是两个大素数，$n=pq$，消息空间和签名空间为 $P=A=Z_n$，密钥空间为：$K=\{(n,p,q,e,d)\mid n=pq,\ ed\equiv 1 \mod \phi(n)\}$，$n,d$ 公开，p,q,e 保密，

则对 $\forall x\in P$，$k=(n,p,q,e,d)\in K$，定义

签名算法为：$sig_k(x)\equiv x^e \mod n$；

验证算法为：$ver_k(x,y)=\text{true} \Leftrightarrow x\equiv y^d \mod n$。

如果攻击者伪造了签名 z，一般地有 $z\not\equiv x^e \mod n$，所以，验证时，有 $z^d\not\equiv(x^e)^d=x^{ed}\equiv x \mod n$，从而 $ver_k(x)=\text{false}$。这就达到了签名的要求。

如果我们既要对消息加密，又要对消息签名，RSA 方法也可胜任。

信源选取签名密钥 $k=(n,p,q,e,d)$，将 n, d 公开，d 作为验证密钥，p,q,e 保密，e 作为签名密钥。

信宿选取加密密钥 $\tilde{k}=\left(\tilde{n},\tilde{p},\tilde{q},\tilde{e},\tilde{d}\right)$，将 $\tilde{n},\tilde{e}$ 公开，$\tilde{e}$ 作为加密密钥公开，$\tilde{p},\tilde{q},\tilde{d}$ 保密，$\tilde{d}$ 作为解密密钥。

不失一般性，设 $n<\tilde{n}$，消息 $x<n$（如果 $x\geqslant n$，则将 x 分成更短的比特串，使每一段都小于 n）。

首先，信源应用自己的签名密钥 e 对消息签名：$y\equiv x^e \mod n$，其次再用信宿的加密密钥 $\tilde{e}$ 加密原消息和签名后的消息：$z\equiv(x\| y)^{\tilde{e}} \mod \tilde{n}$，$(x\| y)$ 表示比特串 x, y 连接成更长的比特串（如果 $x\| y\geqslant\tilde{n}$，则将其分割成更短的组，分组加密），将这最后的消息 z 发送给信宿。

信宿接到消息后，首先用自己的解密密钥 $\tilde{d}$ 解密消息：

$$z^{\tilde{d}}\equiv\left[(x\| y)^{\tilde{e}}\right]^{\tilde{d}}=(x\| y)^{\tilde{e}\tilde{d}}\equiv(x\| y) \mod \tilde{n},$$

其次，应用信源的验证密钥 d 验证消息：$y^d\equiv(x^e)^d=x^{ed}\equiv x \mod n$。

这样就完成了一次消息的签名，加密和解密，验证操作。这里有一点值得注意，签名、加密的顺序可以有先后，那么，哪一种顺序是可取的呢？

假设，先加密一个消息：$y=e_k(x)$，然后再签名这个消息：$z=sig_k(e_{\tilde{k}}(x))$，发送消息（$y,z$）。

这时，攻击者可以用自己的签名来代替原签名：$z'=sig_{\bar{k}}(y)$，把伪造的签名消息（y,z'）发送给信宿，而信宿可以（只可以）用伪造者的验证算法来验证，所以他认为消息来自伪造者，这样伪造者达到了扰乱验证信息来源的目的。所以，先加密后签名的顺序是不可取的。

这里仅以 RSA 为例说明了既加密又签名的算法，事实上，所有公钥体制都可以用作这样的算法。

6.1.3 ElGamal 签名方案

ElGamal 签名方案由 T.ElGamal 于 1985 年提出，改进后，被美国国家标准和技术研究所（NIST）采纳为数字签名标准。

1. ElGamal 签名方案的定义

设 p 是一个素数，Z_p 中的离散对数问题是难处理的，$\alpha\in Z_p^*$ 是一个生成元，$P=Z_p^*$，

$A = Z_p^* \times Z_{p-1}$，令：$K = \left\{(p,\alpha,e,\beta)\text{: } \beta = \alpha^e \bmod p\right\}$，

公开密钥：p,α,β 公开；

秘密密钥：e。

签名算法为：

$$sig_k(x,t) = (\gamma,\delta)\text{，} \gamma = \alpha^t \bmod p\text{，} \delta = (x - e\gamma)t^{-1} \bmod p-1\text{；}$$

其中：$t \in Z_{p-1}^*$ 是一个保密的随机整数，$k = (p,\alpha,e,\beta)$。

验证算法为：

对任意消息及签名 $(x,(\gamma,\delta))$，$x,\gamma \in Z_p^*$，$\delta \in Z_{p-1}$，

$$ver_k(x,(\gamma,\delta)) = \begin{cases} true, & \beta^\gamma \gamma^\delta \equiv \alpha^x \bmod p \\ false, & \beta^\gamma \gamma^\delta \not\equiv \alpha^x \bmod p \end{cases}$$

验证的原理是：如果 (γ,δ) 是 x 的签名，则 $\delta = (x - e\gamma)t^{-1} \bmod p-1$，即有：$e\gamma + t\delta \equiv x \bmod p-1$，从而，

$\beta^\gamma \gamma^\delta \equiv (\alpha^e)^\gamma (\alpha^t)^\delta \equiv \alpha^{e\gamma + t\delta} \equiv \alpha^x \quad \bmod p-1$，于是，得到验证。

2．示例

设 $p = 43$，$\alpha = 3$，$e = 15$，则

$$\beta \equiv \alpha^e \bmod p \equiv 3^{15} \bmod 43 \equiv 22 \bmod 43\text{。}$$

假设我们要签名一个消息 $x = 38$，那么选取随机整数 $t = 37$，则有：$t^{-1} \equiv 25 \bmod 42$。

$$\gamma = \alpha^t = 3^{37} \equiv 20 \bmod 43\text{，}$$

$$\delta = (x - e\gamma)t^{-1} = (38 - 15 \times 20) \times 25 \equiv 2 \bmod 42\text{，}$$

$(\gamma,\delta) = (20,2)$ 就是签名的消息。

验证时，只需作如下计算：

$$22^{20}20^2 \equiv 17 \bmod 43\text{，和 } 3^{38} \equiv 17 \bmod 43\text{，}$$

就可验证签名的真实性。

3．ElGamal 签名的安全性分析

（1）由于密钥 e 是保密的，所以，攻击者要得到这个密钥，必须求解离散对数问题：$e = \log_\alpha \beta$，这是一个困难问题。但是，秘密随机数 t 一旦暴露，则解：

$$e = (x - t\gamma)\delta^{-1} \bmod p-1\text{，}$$

即可求得密钥 e，方案被攻破。

（2）关于秘密随机数 t 的另一个问题是：不可用同一个 t 作两次签名，否则，假设用 t 签名两个消息：$(x_1,(\gamma,\delta_1))$，$(x_2,(\gamma,\delta_2))$，则有：

$$\beta^\gamma \gamma^{\delta_1} \equiv \alpha^{x_1} \bmod p$$

$$\beta^\gamma \gamma^{\delta_2} \equiv \alpha^{x_2} \bmod p$$

得：$\gamma^{\delta_1 - \delta_2} \equiv \alpha^{x_1 - x_2} \bmod p$，将 $\gamma \equiv \alpha^t \bmod p$ 代入得：$\alpha^{t(\delta_1 - \delta_2)} \equiv \alpha^{x_1 - x_2} \bmod p$，于是得：$t(\delta_1 - \delta_2) \equiv x_1 - x_2 \bmod p-1$。

设 $d=\gcd\left(\delta_1-\delta_2, p-1\right)$，则可证明：$d\mid(x_1-x_2)$。

令：$x'=\dfrac{x_1-x_2}{d}, \delta'=\dfrac{\delta_1-\delta_2}{d}, p'=\dfrac{p-1}{d}$，

则得到：$x'\equiv t\delta' \bmod p'$，$\gcd(\delta',p')=1$，进一步得：$t\equiv x'\delta'^{-1} \bmod p'$。这样，$t$ 的值只可能是：$t\equiv x'\delta'^{-1}+lp' \bmod p$，$l=0,1,\cdots,d-1$，可见，$t$ 有 d 个可能的值，用条件：$\gamma\equiv\alpha^t \quad \bmod p$ 即可确定 t 的值，这样就得到了秘密随机数 t，系统也就被攻破。

（3）攻击者为了伪造一个给定消息 x 的签名 (γ,δ)，首先选取 γ，再企图寻找相应的 δ，则他必须从 $\beta^\gamma\gamma^\delta\equiv\alpha^x \bmod p$ 出发来求解 δ，于是，他必须求解离散对数问题 $\delta=\log_\gamma(\alpha^x\beta^{-\gamma})$，同样，他不能如愿。

（4）再一种攻击方法是：攻击者可能伪造一个签名的随机消息，使一个随机消息成为一个合法签名消息，以达到扰乱通信，破坏签名者信誉的目的。他可同时选取 γ，δ，并企图从中解出 x，制造合法签名消息 $(x,(\gamma,\delta))$，当然，x 是一个“随机消息”，但是，他必须从 $\beta^\gamma\gamma^\delta\equiv\alpha^x \bmod p$ 出发来求解 x，这样，他必须解离散对数问题：$x=\log_\alpha\left(\beta^\gamma\gamma^\delta\right)$。

（5）第五种可能的攻击是：同时选取 γ,δ,x。

设 i，j 是整数，$0\leqslant i\leqslant p-2$，$0\leqslant j\leqslant p-2$，$\gcd\ (j,p-1)=1$，令：

$$\gamma=\alpha^i\beta^j \bmod p,\quad \delta=-\gamma j^{-1} \bmod p-1,\quad x=-\gamma i j^{-1} \bmod p-1,$$

其中 j^{-1} 是用模 $p-1$ 计算，$\gcd\ (j,p-1)=1$。

这个 $(x,(\gamma,\delta))$ 就是一个合法签名，但是，消息 x 是随机的。事实上，

$$\begin{aligned}
\beta^\gamma\gamma^\delta &\equiv \beta^{\alpha^i\beta^j}\left(\alpha^i\beta^j\right)^{-\gamma j^{-1}} \bmod p\\
&\equiv \beta^{\alpha^i\beta^j}\left(\alpha^i\beta^j\right)^{-\alpha^i\beta^j j^{-1}} \bmod p\\
&\equiv \beta^{\alpha^i\beta^j}\alpha^{-ij^{-1}\alpha^i\beta^j}\beta^{-\alpha^i\beta^j} \bmod p\\
&\equiv \alpha^{-ij^{-1}\alpha^i\beta^j} \bmod p\\
&\equiv \alpha^{-\gamma ij^{-1}} \bmod p\\
&\equiv \alpha^x \bmod p\text{。}
\end{aligned}$$

可见，这是一个有效签名。

（6）第六种可能的攻击是从一个有效签名出发来伪造一个随机但有效的签名。

设 $(x,(\gamma,\delta))$ 是一个有效签名，h,i,j 是整数，$0\leqslant h,i,j\leqslant p-2$，$\gcd\ (h\gamma-j\delta,p-1)=1$，令：

$$\begin{aligned}
\tilde\gamma &= \gamma^h\alpha^i\beta^j \bmod p\\
\tilde\delta &= \delta\tilde\gamma(h\gamma-j\delta)^{-1} \bmod p-1\\
\tilde x &= \tilde\gamma(hx+i\delta)(h\gamma-j\delta)^{-1} \bmod p-1
\end{aligned}$$

其中 $(h\gamma-j\delta)^{-1}$ 用模 $p-1$ 计算。则可以验证：$\beta^{\tilde\gamma}\tilde\gamma\equiv\alpha^{\tilde x} \bmod p$，可见，$(\tilde x,(\tilde\gamma,\tilde\delta))$ 是一个有效但随机的签名。

6.1.4 数字签名标准 DSS

数字签名算法（Digital Signature Arithmetic，DSA）是 ElGamal 签名方案的改进，《联邦记事》（FIPS）于 1994 年 5 月 19 日公布，同年 12 月 1 日宣布它被采纳为数字签名标准（Digital Signature Standard，DSS）。

1．DSA 的描述

设 p 是一个 512 比特的素数，Z_p^* 中离散对数问题是难处理的。q 是一个 160 比特的素数，$q|p-1$，$\alpha \in Z_p^*$，是模 p 的 q 单位根。令：$P=Z_p^*$, $A=Z_q \times Z_q$，$K=\{(p,q,\alpha,e,\beta)\text{: } \beta=\alpha^e \bmod p\}$

公开密钥：p,q,α,β 公开；

秘密密钥：e 保密。

对 $k=(p,q,\alpha,e,\beta)$，选取一个随机数 t，$1 \leqslant t \leqslant q-1$，并保密。

签名算法：

$$sig_k(x,t)=(\gamma,\delta),$$

其中 $\gamma \equiv (\alpha^t \bmod p) \bmod q$，$\delta \equiv (x+e\gamma)^{t^{-1}} \bmod q$。

对签名消息 (x,γ,δ)，$x \in Z_p^*$，$\gamma,\delta \in Z_q$，

验证算法：

$$Ver_k(x,\gamma,\delta)=\text{true} \Leftrightarrow (\alpha^u \beta^v \bmod p) \bmod q \equiv \gamma,$$

其中 $u \equiv x\delta^{-1} \bmod q$，$v \equiv \gamma\delta^{-1} \bmod q$。

可完全类似地说明验证的数学依据。

2．数字签名标准与 ElGamal 签名方案的差别

ElGamal 签名方案的安全性与离散对数问题的安全性相同，因而，方案中使用的模 p 必须是大素数，至少是 512 比特，有人提议为 1024 比特。然而，512 比特的模素数将导致 1024 比特的签名，这对象灵巧卡之类的使用无疑是一个难题。而 DSS 使用 512 比特的模素数，签名一个 160 比特的消息，产生一个 320 比特的签名。也就是说，模素数不变，但签名的消息比特数少了。离散对数是基于 Z_p^* 的某个阶为 2^{160} 的子群的，从而如果离散对数是安全的，那么系统也是安全的。

选择签名密钥时须注意以下几点：

（1）要选择 q，满足：$q|p-1$；

（2）要选择 α 是模 p 的 q 次单位根，所以，如果令 g 是 Z_p^* 中的一个 q 阶元，则可令 $\alpha=g^{\frac{p-1}{q}}$；

（3）签名运算中要求 $\delta^{-1} \bmod q$，从而，必有 $\delta \not\equiv 0 \bmod q$。而 $\delta \equiv 0 \bmod q$ 发生的概率是 $\frac{1}{2^{160}}$，所以，这种情况是几乎不可能发生的。

（4）由于签名运算中要求 $t^{-1} \bmod q$，从而选择 t 时，要满足：$(t,q)=1$。

3. DSS 示例

令 $p=3011$, $q=301$, $\alpha=114$

设秘密密钥 $e=68$，消息为 $x=2310$，秘密随机数 $t=71$。

$$\beta=114^{68}\equiv 1993 \bmod 3011,$$

$$t^{-1}=71^{-1}\equiv 106 \bmod 301,$$

$$\gamma=\left(114^{71} \bmod 3011\right) \bmod 301\equiv 216,$$

$$\delta=(2310+68\times 216)\times 106 \bmod 301\equiv 2。$$

于是签名密钥为：$k=(3011,301,114,68,1993)$ 和秘密随机数：$t=71$。将 $p=3011$, $q=301$, $\alpha=114$, $\beta=1993$ 公开；$e=68$, $t=71$ 保密。发送签名消息：

计算签名：$sig_k(2310,71)=(\gamma,\delta)=(216,2)$。

验证签名：

$$\delta^{-1}=2^{-1}\equiv 151 \bmod 301,$$

$$e_1=2310\times 151\equiv 252 \bmod 301,$$

$$e_2=216\times 151\equiv 108 \bmod 301,$$

$$\left(114^{252}1993^{108} \bmod 3011\right) \bmod 301=216=\gamma。$$

6.1.5 其他数字签名方案

DSS 和 DES 一样，人们一直认为其中有陷门，Gus Simmons 发现了其中的阈下信道。所以一些改进方案被提了出来。

除了以上介绍的数字签名方案以外，日本、欧洲等地学者也提出了许多优秀的签名方案，并得到了广泛的应用。

如：日本 NTT 提出的 ESIGN 数字签名方案，Ong-Schnorr-Shamir 签名方案。

数字签名不但可以用于简单的签名要求，还可以应用于特别的数字签名要求。如：盲签名、不可否认签名、多重签名、团体签名、一次签名、失败—终止签名等等。下面仅就盲签名和不可否认签名进行讨论。

1. 盲签名

一般地，签名者总是首先要知道签名文件内容，然后才会对文件签名。但是，有时候，签名者并不需要对文件内容负责，而仅仅需要证明文件内容的真实性。我们用预测魔术来说明这种需要。

魔术师将预测的数字写在一张纸片上折叠后，请一个观众作公证人，将纸片放入信封封闭后交给助手。然后，魔术师再请任意一个观众选择一个数字，并煞有介事地当众打开信封取出纸片，宣布他确实早已预测了这个数字。

我们注意到公证人的工作，他证明数字纸片的事实，但并不知道预测的数字（其实纸片上可能什么也没有写），这就是盲签名的思想。但遗憾的是，公证人并没有真正履行签名的职责，因为他没有在信封上签名，所以魔术师的助手轻松地偷换信封。如果公证人能够真正地在信封上签名，打开信封时能够仔细地验证签名，那么，魔术师就会原形毕露了。

这种签名方法是借助于信封来隐蔽信息的，似乎与传统的签名有较大的差距，其实，我

们也完全可以用传统的方法来实现，而不必借助于信封，在数字纸片上盖上复写纸，再盖上白纸，公证人在白纸上签上自己的名字，这时纸片也就有了公证人的签字。需要时，拿出签字即可验证。

盲签名的基本原理是两个可换的加密算法的应用，第一个加密算法是为了隐蔽信息，可称为盲变换，第二个加密算法才是真正的签名算法。

设文件 x 的拥有者为 A，签名人为 B。盲算法（及其密钥）E_{k_a} 为 A 拥有，签名算法（及其密钥）E_{k_b} 为 B 拥有，所有算法及验证密钥公开，其他密钥保密，则盲签名过程为：

（1）A 用盲变换将文件 x 加密为 $c_a=E_{k_a}(x)$ 后发送给 B；

（2）B 将信息 $c_a=E_{k_a}(x)$ 加密为 $c_{ab}=E_{k_b}(E_{k_a}(x))$ 后发还给 A；

（3）A 解盲 $c_{ab}=E_{k_b}(E_{k_a}(x))$ 得到：

$$c_b=D_{k_a}(E_{k_b}(E_{k_a}(x)))=E_{k_b}(D_{k_a}(E_{k_a}(x)))=E_{k_b}(x)$$

A 发布 B 的签名消息：$(c_a,\ c_{ab})$。

验证时，A 公布消息（x，c_b），验证：$c_{ab}=E_{k_b}(c_a)$。

盲变换要根据具体的签名算法而定，基本的要求是它与签名算法必须可换。

1985 年，D.Chaum 给出了一个 RSA 的盲签名算法：

设 B 的签名密钥为 e，验证密钥为 d，模数为 n。A 有信息 m，需要 B 对 m 进行盲签名。

（1）A 选取一个随机数 $1<k<m$，作盲变换：

$$t\equiv mk^d \bmod n$$

发送给 B；

（2）B 对 t 签名：$t^e\equiv(mk^d)^e \bmod n$

发送给 A；

（3）A 解盲：$S\equiv t^e/k \bmod n=m^e \bmod n$。

显然盲变换主体为模 n 乘法，与模幂运算可换。

2．不可否认签名

1989 年 Chaum 和 Antwerpen 引入不可否认签名。方案的本质是签名的验证必须有签名者合作才能完成，但签名者又不能否认签名。这一方案的目的是阻止签名文件的随便复制、任意散布。

例如：A 向 B 借钱，A 必须在借条上签字。也许有一天，B 提前向 A 索要钱款，或者 A、B 之间发生了矛盾，为此，B 复制了 A 的借条到处散布，并验证给人们看，A 确实借了 B 的钱，使 A 的生意受到了巨大的损失。这是 A 不希望发生的，所以，A 希望借条必须有他的合作才能验证。

没有（不能验证）签名的借条，等于废纸一张，因而没有人会相信 B 展示的借条，A 的设想确实达到了他原来的目的。

但是，B 会担心 A 任何时候都不合作，即使在法官面前也不合作，从而就没办法验证借条的真实性，而讨回借款了。

这样一对矛盾的要求如何实现呢？

不可否认签名，借用了法律的原理来实现这一目标。不是去证明 B 的事实，而是去证明 A 的事实。

从 A 的角度分析问题：

（1）如果 A 拒绝合作，则认为 B 提供的是事实；

（2）如果 A 用虚假方法验证，算法可以发现 A 的行为，也判定 B 提供的是事实。

（3）如果 B 提供的借条是伪造的，则 A 也可以发现并证明，从而否认 B 提供的借条。

所以，不可否认签名由三个部分组成：签名算法、验证算法及不可否认算法。

下面给出 Chaum-Antwerpen 不可否认签名算法。

（1）算法描述

设 A 是签名者，B 是信宿。

令 q，$p=2q+1$ 都是素数，求 Z^*_p 中的离散对数是困难的。$1\leqslant e,d\leqslant q-1$，$ed\equiv 1 \bmod q$；

a 是 Z^*_p 中的一个 q 阶元素（$a^q=1$，$a^t\neq 1$，$1<t<q$），$b\equiv a^e \bmod p$；

A 发布公开信息：p，a，b；(e, d)为签名、验证密钥，都保密；

1）签名算法

设待签名信息为 $m\in Z^*_q$，则 A 计算签名：

$$c=sig_e(m)\equiv m^e \bmod p;$$

2）验证算法

对于签名信息（m，c），

a）B 选择随机数 e_1，$e_2\in Z^*_p$，计算 $\tilde{c}\equiv c^{e_1}b^{e_2} \bmod p$，发送给 A；

b）A 计算 $\tilde{\tilde{c}}\equiv \tilde{c}^d \bmod p$，发送给 B；

c）B 验证：$ver(m,c)=\begin{cases}\text{True}, & \tilde{\tilde{c}}\equiv m^{e_1}a^{e_2} \bmod p \\ \text{False}, & \text{其他}\end{cases}$；

3）不可否认算法

首先：

a）B 选择随机数 e_1，$e_2\in Z^*_p$，计算 $\tilde{c}\equiv c^{e_1}b^{e_2} \bmod p$；

b）A 计算 $\tilde{\tilde{c}}\equiv \tilde{c}^d \bmod p$，

c）B 验证：$\tilde{\tilde{c}}\equiv m^{e_1}a^{e_2} \bmod p$。

再一次：

a）B 再选择随机数 f_1，$f_2\in Z^*_p$，计算 $\tilde{g}\equiv c^{f_1}b^{f_2} \bmod p$；

b）A 计算 $\tilde{\tilde{g}}\equiv \tilde{g}^d \bmod p$；

c）B 验证：$\tilde{\tilde{g}}\equiv m^{f_1}a^{f_2} \bmod p$。

如果签名人 A 的合作是真诚的，则一定可以通过一致性检验：

$\left(\tilde{\tilde{c}}a^{-e_2}\right)^{f_1}\equiv\left(\tilde{\tilde{g}}a^{-f_2}\right)^{e_1} \bmod p$，否则 B 就可以发现 A 的违约行为。

（2）验证原理及不可否认原理

1）验证原理

如果（m，c）是一个合法签名，则：

$$\tilde{\tilde{c}}=\tilde{c}^d=\left(c^{e_1}b^{e_2}\right)^d \bmod p$$

$$=c^{e_1 d}b^{e_2 d} \bmod p=\left(m^e\right)^{e_1 d}\left(a^e\right)^{e_2 d} \bmod p$$

$= m^{ee_1 d} a^{ee_2 d} \bmod p = m^{e_1} a^{e_2} \bmod p$。

2）不可否认原理

$\left(\tilde{\tilde{c}} a^{-e_2}\right)^{f_1} \bmod p$

$\equiv \left(\tilde{c}^d a^{-e_2}\right)^{f_1} \bmod p \equiv \left(\left(c^{e_1} b^{e_2}\right)^d a^{-e_2}\right)^{f_1} \bmod p \equiv \left(c^{e_1 d} b^{e_2 d} a^{-e_2}\right)^{f_1} \bmod p$

$\equiv c^{e_1 f_1 d} b^{e_2 f_1 d} a^{-e_2 f_1} \bmod p \equiv m^{ee_1 f_1 d} a^{ee_2 f_1 d} a^{-e_2 f_1} \bmod p$

$\equiv \left(m^{ed}\right)^{e_1 f_1} \left(a^{ed-1}\right)^{e_2 f_1} \bmod p$。

$\left(\tilde{\tilde{g}} a^{-f_2}\right)^{e_1} \bmod p$

$\equiv \left(\tilde{g}^d a^{-f_2}\right)^{e_1} \bmod p \equiv \left(c^{f_1 d} b^{f_2 d} a^{-f_2}\right)^{e_1} \bmod p$

$\equiv \left(m^{ed}\right)^{e_1 f_1} \left(a^{ed-1}\right)^{e_1 f_2} \bmod p$

可见，$\left(\tilde{\tilde{c}} a^{-e_2}\right)^{f_1} \equiv \left(\tilde{\tilde{g}} a^{-f_2}\right)^{e_1} \bmod p \Leftrightarrow \left(a^{ed-1}\right)^{e_2 f_1} \equiv \left(a^{ed-1}\right)^{e_1 f_2} \bmod p$，此与具体信息无关，如果 $ed \equiv 1 \bmod p$，则可以通过一致性验证。

反之，如果 $ed \not\equiv 1 \bmod p$，通过一致性验证的可能性是很小的。

所以，一致性检验不但可以检查签名者的非法验证行为，也可以证明其验证行为的真实性，因而它也可以检查伪造签名。

6.2 散列函数

6.2.1 签名与散列函数

前面讲的数字签名方案一般都是对一个较短的消息签名而言，对于一个很长的消息，如果直接处理，则首先要把它分割成较短的段，再进行签名。显然，这种处理非常麻烦，不实用。例如，要签名一个几兆的文件，以前的处理方法是不合适的。

解决这一问题的办法是：对需要签名的消息用一个函数，产生一个固定长度的消息摘要（Message Digest），最后对消息摘要进行签名，得到确定长度的签名消息。

这种函数称为散列（Hash）函数。所谓散列函数就是消息空间到另一个消息空间的一个映射。一般地，函数值长度固定，即函数值的比特数固定，比如 160 比特。

定义 6-1 如果函数 h 满足下面两个性质：

（1）压缩性：h 映任意有限长度的输入 x，为固定长度 n 的输出 $h(x)$；

（2）易计算：给定输入 x，$h(x)$是易计算的；

则称 h 是一个散列函数。

显然，将消息对应到散列函数值是一个多对一的映射，所以，可能出现多个不同消息具有相同散列函数值的情形，这种现象称为碰撞。

如前所述，签名是对消息的散列函数值签名，那么，消息的碰撞也就意味着存在伪造签名的可能。所以，散列函数应该是一个计算上不可逆的函数，即：要选取不同两个消息具有

相同散列函数值，或者，选取另一个消息使具有给定（消息的）散列函数值是计算上不可能的。为此，提出了无碰撞散列函数的概念。

6.2.2 无碰撞散列函数

攻击者一个可能的一个攻击方法是：设 (x,y) 是一个有效的签名 (x,y)，$h(x)$ 是消息 x 的散列函数值，$y=sig_k(h(x))$ 是对散列值 $h(x)$ 的签名。如果攻击者找到一个消息 $x'\neq x$，$h(x')=h(x)$，则 (x',y) 是一个有效签名，但它是一个伪造签名。当然，一般地，x' 只能是一个随机消息，不过，攻击者达到了干扰通信的目的。为了阻止这种攻击，散列函数必须满足下面定义的弱无碰撞性：

定义 6-2 如果对给定的一个消息 x，找到一个满足 $x'\neq x$，$h(x')=h(x)$ 的消息 x 是计算上不可能的，则称散列函数 h 是弱无碰撞的。

另一个可能的攻击是：攻击者事先找到两个消息 $x'\neq x$，$h(x')=h(x)$，并骗得签名者对消息 x 的签名，则 $\left(x',h\left(x'\right)\right)=\left(x',h\left(x\right)\right)$ 就是一个合法的，伪造签名。

为了阻止这种伪造，需要更强的无碰撞性。

定义 6-3 如果找到两个满足 $x'\neq x$，$h(x')=h(x)$ 的消息 x，x'是计算上不可能的，则称散列函数 h 是强无碰撞的。

显然，如果一个散列函数是强无碰撞的，则该函数一定是弱无碰撞的。

对于以上两类攻击，有一个较形象的描述，称为生日攻击。这是一个典型的概率问题：一群人中，至少有一人与你的生日相同的概率大于 $\frac{1}{2}$，那么这群人至少有多少人？结论是183。一群人中至少有两人生日相同的概率大于 $\frac{1}{2}$，那么这群人至少有多少人？结论是 23。这两种状况恰恰对应于上面的弱、强碰撞现象。寻找具有某个特定生日的人类似于第一种碰撞，寻找两个具有相同生日的人类似于第二种碰撞。

现实中，可以应用如下方法伪造签名：伪造者准备好两个不同内容的文件（可能仅在很少的关键字上有差别，但利害却相反），一个对他有利，记为版本 A，一个对你有利，记为版本 B。显然你只会给对你有利的版本 B 的散列值签名，记签名为：$\left(B,Sig\left(h\left(B\right)\right)\right)$。

伪造者采用如下方法对文件 A 作细微的修改，如：在回车之前加或不加空格，如果文件有 32 行，则在这 32 行行尾加或不加空格可以得到文件的 2^{32} 种不同版本 A_i（但文件意义并没有改变，且修改也很难发现），计算这 2^{32} 个版本的散列值 $h(A_i)$，根据第一类生日攻击，如果输出的散列值比特长度不大，那么从 $h(A_i)$ 中找到 $h(B)$ 是可行的，例如有：$h\left(A_{i_0}\right)=h\left(B\right)$，则 $\left(A_{i_0},Sig\left(h\left(A_{i_0}\right)\right)\right)=\left(B,Sig\left(h\left(A_{i_0}\right)\right)\right)$ 就是一个合法但伪造的签名。

如果对 A、B 同时作以上的修改，得到两类文件 $A_i,B_j\left(i,j=1,2,\cdots,2^{32}\right)$，根据第二类生日攻击，找到 $h\left(A_i\right)=h\left(B_j\right)$ 的概率更大。如果有 $h\left(A_{i_0}\right)=h\left(B_{j_0}\right)$，则 $\left(B_{j_0},Sig\left(h\left(A_{j_0}\right)\right)\right)$ $=\left(A_{j_0},Sig\left(h\left(A_{j_0}\right)\right)\right)$ 就是一个合法的伪造签名。

这种攻击的危害性是显然的，所以，选用的散列函数不但应具有弱无碰撞性，而且还应具有强无碰撞性。

当然，以上的做法是计算上不可能的。

另一种可能的攻击是对散列函数的计算上不可逆的攻击，于是有下面的定义：

定义 6-4 如果对给定一个消息摘要 z，找到一个消息 x，满足 $z=h(x)$ 是计算上不可能的，则称散列函数具有单向性。

例如：设 $n=pq$， p,q 是素数，则适当地选择 p,q， $f(x)=(x^2 \bmod n)$ 是一个单向函数。

选择安全的加密算法 E 及适当的密钥 k，对消息 x 加密得： $y=E_k(x)$，则已知 y，求 x 是计算上不可能的，所以这也是一个单向函数。

攻击者选择一个随机消息摘要 z，并设法计算出签名 $y=sig_k(z)$，这时，如果他能找到一个消息 x，满足 $z=h(x)$，则 (x,y) 是一个合法伪造签名。如果散列函数 h 具有单向性，则可有效地阻止这种攻击。

可以证明强无碰撞性包含了单向性。事实上，如果一个散列函数不具单向性，而具有一个逆算法，则存在一个寻找碰撞的概率算法 Las Vegas 算法，它能找到一个碰撞的概率至少为$\frac{1}{2}$。

可见，强无碰撞性是一个很强的概念，它不但包含了弱碰撞性，又包含了单向性。所以，安全的散列函数应该是强无碰撞的。

6.2.3 离散对数散列函数

Chaum-van Heijst-Pfitzmann 提出一个基于离散对数的散列函数，其描述如下：

假设 $p=2q+1$ 是一个大素数，其中 q 也是一个素数。设 α 和 β 是 Z_p^* 中的两个生成元，离散对数 $\log_\alpha \beta$ 的计算是困难的，将α，β保密。

散列函数

$$h\colon\ Z_q \times Z_q \to Z_p^*$$

定义为：

$$h(x_1,x_2)=\alpha^{x_1}\beta^{x_2} \bmod p\text{。}$$

下面定理表示只要离散对数是安全的，则这个散列函数是安全的，即散列函数是强无碰撞的。

定理 6-1 如果能找到 Chaum-van Heijst-Pfitzmann 散列函数 h 的一个碰撞，则离散对数 $\log_\alpha \beta$ 可有效计算。

证明 设 $(x_1,x_2)\neq(y_1,y_2)$， $h(x_1,x_2)=h(y_1,y_2)$ 是一个碰撞，则有：

$\alpha^{x_1}\beta^{x_2} \equiv \alpha^{y_1}\beta^{y_2} \bmod p$，于是，

$$\alpha^{x_1-y_1} \equiv \beta^{y_2-x_2} \bmod p\text{，} \tag{6-1}$$

令 $d=\gcd\ (y_2-x_2,p-1)=\gcd\ (y_2-x_2,2q)$，而 q 是素数，所以，必有 $d\in\{1,2,q,p-1\}$。

下面分别讨论。

（1） $d=1$时，可知$(y_2-x_2)^{-1} \bmod p-1$存在，从而有：

$$\beta \equiv \beta^{(y_2-x_2)(y_2-x_2)^{-1}} \equiv \alpha^{(x_1-y_1)(y_2-x_2)^{-1}} \bmod p,$$

于是得离散对数，$\log_\alpha \beta = (x_1-y_1)(y_2-x_2)^{-1} \bmod p-1$。

（2）$d=2$ 时，可知 y_2-x_2 是偶数，而 q 是奇素数，所以，$\gcd(y_2-x_2, q)=1$。

于是 $(y_2-x_2)^{-1} \bmod q$ 存在，所以，存在正整数 k，$(y_2-x_2)(y_2-x_2)^{-1}=kq+1$。
又由于 β 是生成元，所以，$\beta^{p-1} \equiv 1 \bmod p$，$\beta^t \not\equiv 1 \bmod p$，$0<t<p-1$，

但是：$(\beta^q)^2 \equiv \left(\beta^{\frac{p-1}{2}}\right)^2 \equiv \beta^{p-1} \equiv 1 \bmod p$，所以，$\beta^q \equiv -1 \pmod p$。

同理，$\alpha^q \equiv -1 \pmod p$。

于是有：

$$\begin{aligned}\beta^{(y_2-x_2)(y_2-x_2)^{-1}} &\equiv \beta^{kq+1} \bmod p\\ &\equiv (-1)^k \beta \bmod p\\ &\equiv \pm\beta \bmod p,\end{aligned}$$

于是由式（6-1）得：

$$\begin{aligned}\alpha^{(x_1-y_1)(y_2-x_2)^{-1}} &\equiv \beta^{(y_2-x_2)(y_2-x_2)^{-1}} \bmod p\\ &\equiv \pm\beta \bmod p\end{aligned}$$

如果 $\alpha^{(x_1-y_1)(y_2-x_2)^{-1}} \equiv \beta \bmod p$，则离散对数 $\log_\alpha \beta$ 为：

$\log_\alpha \beta = (x_1-y_1)(y_2-x_2)^{-1} \bmod p-1$；

如果 $\alpha^{(x_1-y_1)(y_2-x_2)^{-1}} \equiv -\beta \bmod p$，则：

$$-\alpha^{(x_1-y_1)(y_2-x_2)^{-1}} \equiv \beta \bmod p-1,$$

即：$\alpha^q \alpha^{(x_1-y_1)(y_2-x_2)^{-1}} \equiv \alpha^{(x_1-y_1)(y_2-x_2)^{-1}+q} \equiv \beta \bmod p-1$，

所以：$\log_\alpha \beta = (x_1-y_1)(y_2-x_2)^{-1}+q \bmod p-1$）。

于是，只需验证：$\alpha^{(x_1-y_1)(y_2-x_2)^{-1}} \equiv \beta \bmod p-1$ 和

$$\alpha^{(x_1-y_1)(y_2-x_2)^{-1}+q} \equiv \beta \bmod p-1$$

就可确定 $\log_\alpha \beta$。

（3）$d=q$ 时，因为：$0 \leqslant x_2 \leqslant q-1, 0 \leqslant y_2 \leqslant q-1$，
所以，$-(q-1) \leqslant y_2-x_2 \leqslant q-1$，从而不可能有 $d=q$。

（4）$d=p-1$，必有：$y_2=x_2$，则有：

$$\alpha^{x_1}\beta^{y_2} \equiv \alpha^{y_1}\beta^{y_2} \bmod p,\quad \alpha^{x_1} \equiv \alpha^{y_1} \bmod p,$$

得：$x_1=y_1$，于是：$(x_1,x_2)=(y_1,y_2)$，与假设矛盾，所以，$d=p-1$ 也是不可能的。
定理证毕。

【例 6-1】 设 $q=6173$，$p=12347=2\times6173+1$，$\alpha=2$，$\beta=8461$。

已知碰撞：$(5692,144) \neq (212,4214)$，$\alpha^{5692}\beta^{144} \equiv \alpha^{212}\beta^{4214} \bmod 12347$，求离散对数 $\log_\alpha \beta$。

解 因为：$\gcd(y_2-x_2, p-1)=2$，所以考虑：

（1） $(y_2-x_2)^{-1} \bmod q \equiv (4214-144)^{-1} \bmod 6173 \equiv 4312 \bmod 6173$。

（2） $(x_1-y_1)(y_2-x_2)^{-1} \bmod p-1 \equiv (5692-212)4312 \bmod 12346 \equiv 11862 \bmod 12346$。

而：$\alpha^{(x_1-y_1)(y_2-x_2)^{-1}} \bmod p \equiv 2^{11862} \bmod 12347 = 9998$，$\alpha^{(x_1-y_1)(y_2-x_2)^{-1}+q} \bmod p \equiv 2^{11862+6173} \bmod 12347 \equiv 2^{5689} \bmod 12347 = 8461$

所以，$\log_\alpha \beta = (x_1-y_1)(y_2-x_2)^{-1}+q \bmod p-1 = 5689$。

6.2.4 散列函数的扩展

前面讨论的散列函数具有有限的定义域，显然不能完全满足我们的需要。如：当要签名文本文件时，被签名消息可以是任意长度的。所以，需要具有无限定义域的强无碰撞散列函数。下面讨论将具有有限定义域的强无碰撞散列函数扩展为具有无限定义域的强无碰撞散列函数的方法。

1．散列函数扩展的一般方法

一般地，应用一个$r+m$ 比特输入，m 比特输出的散列函数 f，可以通过以下级联（也称迭代）方法来构造无限长度输入的散列函数 h。

（1）分割消息 M 为长度 r 的分组 M_1，M_2，…，M_t；

（2）对最后的分组进行适当的填充，有时候需要再添加一个分组；

（3）选择一个初始向量 IV；

（4）作迭代计算：$H_i = f(H_{i-1}, M_i)$，$i=1,2,\cdots,t$，$H_0=IV$；

（5）最后输出处理函数 g 是一个 m 比特输入，n 比特输出的函数，如果 $m=n$，通常 g 是恒等映射。

方法如图 6-1 所示。

2．Merkle 级联算法（Merkle-Damgard 构造）

（1）设 f 是一个强无碰撞的散列函数，输入为 $r+m$ 比特，输出为 n 比特，如 $r=128$ 比特，$n=512$ 比特；

（2）设原消息 M 的长度为 b，将 M 分割为长度 r 比特的消息 x_1, x_2, …, x_t，最后一个分组不足 r 比特时，右边用 0 填充；

（3）添加一个辅助分组 x_{t+1}，令其为 b 的二进制表示，不足 r 比特时，高位用 0 填充（假设$b<2^r$）；

（4）迭代计算：$H_0 = \underbrace{0\cdots0}_{n\ bits}$，$H_i = f(H_{i-1},\ x_i)$，$i=1,2,\cdots,t+1$；

（5）定义散列函数：h: $\{0,1\}^\infty \to \{0,1\}^n$，$h(x)=H_{t+1}$。

h 是强无碰撞的证明留作练习。

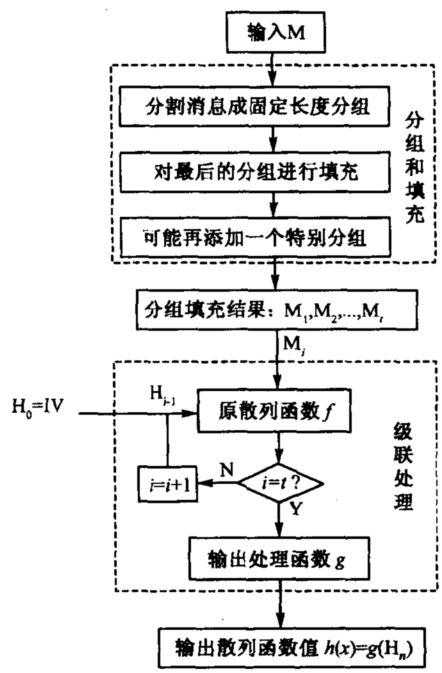


图 6-1 散列函数的一般扩展方法

3．基于分组加密算法的散列函数扩展

如果用一个有效的分组加密算法（如 DES，AES 等）作为散列函数，可以证明它是

强无碰撞的。从而可以用它们作为原始散列函数进行扩展。

E: $\{0,1\}^r \times \{0,1\}^n \to \{0,1\}^n$ 是一个有效的加密算法。

$x = x_1 x_2 \cdots x_t$ 是消息的一个分割和填充。

（1）Matyas-Meyer-Oseas 散列函数

定义：h: $\{0,1\}^\infty \to \{0,1\}^n$，$h(x) = H_t$，

其中：$H_0 = IV$，$H_i = f(H_{i-1}, x_i) = E_{g(H_{i-1})}(x_i) \oplus x_i$，$i = 1, 2, \cdots, t$；

g: $\{0,1\}^n \to \{0,1\}^r$ 是一个适当的映射。如图 6-2 所示。

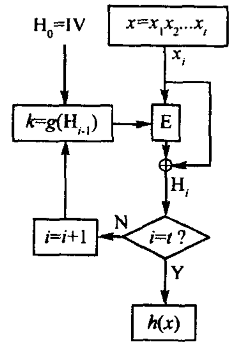


图 6-2　Matyas-Meyer-Oseas 散列函数

（2）Davies-Meyer 散列函数

定义：h: $\{0,1\}^\infty \to \{0,1\}^n$，$h(x) = H_t$。

$$H_0 = IV，\quad H_i = f(H_{i-1}, x_i) = E_{x_i}(H_{i-1}) \oplus H_{i-1}，\quad i = 1, 2, \cdots, t；$$

如图 6-3 所示。

（3）Miyaguchi-Preneel 散列函数

定义：h: $\{0,1\}^\infty \to \{0,1\}^n$，$h(x) = H_t$，

其中：$H_0 = IV$，$H_i = f(H_{i-1}, x_i) = E_{g(H_{i-1})}(x_i) \oplus x_i \oplus H_{i-1}$，$i = 1, 2, \cdots, t$；

g: $\{0,1\}^n \to \{0,1\}^r$ 是一个适当的映射。如图 6-4 所示。

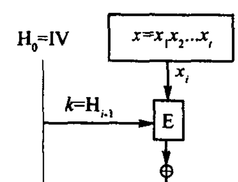


图 6-3　Davies-Meyer 散列函数

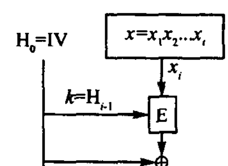


图 6-4　Miyaguchi-Preneel 散列函数

6.3　散列函数标准 SHS

6.3.1　MD5

1990 年 Rivest 提出了 MD（Message Digest 的缩写）系列散列函数 MD4，MD4 输入任意长度的消息，输出 128 比特的消息摘要，是一个强无碰撞的散列函数，适于软件的高速实

现。公布后，接受了大量的密码分析和攻击，虽然还没有被攻破，但是也显示出了算法的弱点，于是 Rivest 于 1991 年提出了其改进算法 MD5。MD4 和 MD5 都得到了广泛的应用，成为许多机构和组织的散列函数标准，如 RFC 2289 等。

MD5 比 MD4 更复杂，但设计思想相似。

1. MD5 基本数据和运算：

（1）分割和填充

1）将消息 x 分割成 512 比特的消息块 $x_1,x_2,\cdots,x_t$，最后一个块填充为：$x_t=\overbrace{*\cdots*10\cdots0\ \underbrace{\text{length}(x)}_{64bits}}^{512\ bits}$，其中 length($x$) 表示 x 的长度的二进制形式，不足 64 比特时高位添一个介符 1 再补 0;

2）将每个消息块划分成 16 个 32 比特的子块 $M_1,\cdots,M_{16}$；

（2）4 个 32 比特的初始向量：

$A_0=01234567$，$B_0=89ABCDEF$，$C_0=EFDCBA98$，$D_0=76543110$。

（3）基本运算和四个基本函数：

1）基本运算：

$X\wedge Y$：X，Y 的逐比特逻辑"与"；

$X\vee Y$：X，Y 的逐比特逻辑"或"；

$X\oplus Y$：X，Y 的逐比特逻辑"异或"；

$\tilde{X}$：X 的逐比特逻辑"反"；

$X<<<s$：X 循环向左移 s 位；

2）基本函数：

$$F(x,y,z)=(x\wedge y)\vee(\tilde{x}\wedge z),$$
$$G(x,y,z)=(x\wedge z)\vee(y\wedge\tilde{z}),$$
$$H(x,y,z)=x\oplus y\oplus z,$$
$$I(x,y,z)=y\oplus(x\vee\tilde{z})。$$

2. MD5 算法

（1）$A=A_0$，$B=B_0$，$C=C_0$，$D=D_0$；

（2）作 4 轮计算（$MD4$ 只有 3 轮），每一轮作 16 次操作。令：

第一轮：

$$A=B+(A+(F(B,C,D)+M_0+7)<<<0xd76aa478)$$
$$D=A+(D+(F(A,B,C)+M_1+12)<<<0xe8c7b756)$$
$$C=D+(C+(F(D,A,B)+M_2+17)<<<0x242070db)$$
$$B=C+(B+(F(C,D,A)+M_3+22)<<<0xc1bdceee)$$

$$A=B+(A+(F(B,C,D)+M_4+7)<<<0xf57c0faf)$$
$$D=A+(D+(F(A,B,C)+M_5+12)<<<0x4787c62a)$$
$$C=D+(C+(F(D,A,B)+M_6+17)<<<0xa8304613)$$
$$B=C+(B+(F(C,D,A)+M_7+22)<<<0xfd469501)$$

$$A = B + (A + (F(B,C,D) + M_8 + 7) <<< 0x698098d8)$$
$$D = A + (D + (F(A,B,C) + M_9 + 12) <<< 0x8b44f7af)$$
$$C = D + (C + (F(D,A,B) + M_{10} + 17) <<< 0xffff5bb1)$$
$$B = C + (B + (F(C,D,A) + M_{11} + 22) <<< 0x8950cd7be)$$

$$A = B + (A + (F(B,C,D) + M_{12} + 7) <<< 0x6b901122)$$
$$D = A + (D + (F(A,B,C) + M_{13} + 12) <<< 0xfd987193)$$
$$C = D + (C + (F(D,A,B) + M_{14} + 17) <<< 0xfa679438e)$$
$$B = C + (B + (F(C,D,A) + M_{15} + 22) <<< 0x49b40821)$$

第二轮：

$$A = B + (A + (G(B,C,D) + M_1 + 5) <<< 0xf61e2562)$$
$$D = A + (D + (G(A,B,C) + M_6 + 9) <<< 0xc040b340)$$
$$C = D + (C + (G(D,A,B) + M_{11} + 14) <<< 0x265e5a51)$$
$$B = C + (B + (G(C,D,A) + M_0 + 20) <<< 0xe9b6c7aa)$$

$$A = B + (A + (G(B,C,D) + M_5 + 5) <<< 0xdb2f105d)$$
$$D = A + (D + (G(A,B,C) + M_{10} + 9) <<< 0x02441453)$$
$$C = D + (C + (G(D,A,B) + M_{15} + 14) <<< 0xdba1e681)$$
$$B = C + (B + (G(C,D,A) + M_4 + 20) <<< 0xe7d3fbc8)$$

$$A = B + (A + (G(B,C,D) + M_9 + 5) <<< 0x21e1cde6)$$
$$D = A + (D + (G(A,B,C) + M_{14} + 9) <<< 0xc33707d6)$$
$$C = D + (C + (G(D,A,B) + M_3 + 14) <<< 0xf4d50d87)$$
$$B = C + (B + (G(C,D,A) + M_8 + 20) <<< 0x455a14ed)$$

$$A = B + (A + (G(B,C,D) + M_{13} + 5) <<< 0xa9e3e905)$$
$$D = A + (D + (G(A,B,C) + M_2 + 9) <<< 0xfcefa3f8)$$
$$C = D + (C + (G(D,A,B) + M_7 + 14) <<< 0x676f02d9)$$
$$B = C + (B + (G(C,D,A) + M_{12} + 20) <<< 0x8d2a4c8a)$$

第三轮：

$$A = B + (A + (H(B,C,D) + M_5 + 4) <<< 0xfffa3942)$$
$$D = A + (D + (H(A,B,C) + M_8 + 11) <<< 0x8771f681)$$
$$C = D + (C + (H(D,A,B) + M_{11} + 16) <<< 0x6d9d6122)$$
$$B = C + (B + (H(C,D,A) + M_{14} + 23) <<< 0xfde5380c)$$

$$A = B + (A + (H(B,C,D) + M_1 + 4) <<< 0xa4beea44)$$
$$D = A + (D + (H(A,B,C) + M_4 + 11) <<< 0x4bdecfa9)$$
$$C = D + (C + (H(D,A,B) + M_7 + 16) <<< 0xf6bb4b60)$$
$$B = C + (B + (H(C,D,A) + M_{10} + 23) <<< 0xbebfbc70)$$

$$A = B + \left(A + \left(H(B,C,D) + M_{13} + 4\right) <<< 0x289b7ec6\right)$$
$$D = A + \left(D + \left(H(A,B,C) + M_0 + 11\right) <<< 0xeaa127fa\right)$$
$$C = D + \left(C + \left(H(D,A,B) + M_3 + 16\right) <<< 0xd4ef3085\right)$$
$$B = C + \left(B + \left(H(C,D,A) + M_6 + 23\right) <<< 0x04881d05\right)$$

$$A = B + \left(A + \left(H(B,C,D) + M_9 + 4\right) <<< 0xd9d4d039\right)$$
$$D = A + \left(D + \left(H(A,B,C) + M_{12} + 11\right) <<< 0xe6db99e5\right)$$
$$C = D + \left(C + \left(H(D,A,B) + M_{15} + 16\right) <<< 0x1fa27cfb\right)$$
$$B = C + \left(B + \left(H(C,D,A) + M_2 + 23\right) <<< 0xc4ac5665\right)$$

第四轮：

$$A = B + \left(A + \left(I(B,C,D) + M_0 + 6\right) <<< 0xf4292244\right)$$
$$D = A + \left(D + \left(I(A,B,C) + M_7 + 11\right) <<< 0x411aff97\right)$$
$$C = D + \left(C + \left(I(D,A,B) + M_{14} + 15\right) <<< 0xab9423a7\right)$$
$$B = C + \left(B + \left(I(C,D,A) + M_5 + 21\right) <<< 0xfc93a039\right)$$

$$A = B + \left(A + \left(I(B,C,D) + M_{12} + 6\right) <<< 0x655b59c3\right)$$
$$D = A + \left(D + \left(I(A,B,C) + M_3 + 10\right) <<< 0x8f0ccc92\right)$$
$$C = D + \left(C + \left(I(D,A,B) + M_{10} + 15\right) <<< 0xffeff47d\right)$$
$$B = C + \left(B + \left(I(C,D,A) + M_1 + 21\right) <<< 0x85845dd1\right)$$

$$A = B + \left(A + \left(I(B,C,D) + M_8 + 6\right) <<< 0x6fa87e4f\right)$$
$$D = A + \left(D + \left(I(A,B,C) + M_{15} + 10\right) <<< 0xfe2ce6e0\right)$$
$$C = D + \left(C + \left(I(D,A,B) + M_6 + 15\right) <<< 0xa3014314\right)$$
$$B = C + \left(B + \left(I(C,D,A) + M_{13} + 21\right) <<< 0x4e0811a1\right)$$

$$A = B + \left(A + \left(I(B,C,D) + M_4 + 6\right) <<< 0xf7537e82\right)$$
$$D = A + \left(D + \left(I(A,B,C) + M_{11} + 10\right) <<< 0xbd3af235\right)$$
$$C = D + \left(C + \left(I(D,A,B) + M_2 + 15\right) <<< 0x2ad7d2bb\right)$$
$$B = C + \left(B + \left(I(C,D,A) + M_9 + 21\right) <<< 0xeb86d391\right)$$

其中 $L <<< c_i (1 \leqslant i \leqslant 16 \times 4)$ 是将 L 循环左移 c_i 位，c_i 是第 i 次操作中的移位次数，它是 $2^{32} \times |\sin(i)|$ 的整数部分，i 是弧度。

（3）将 A_0, B_0, C_0, D_0 分别加到 A, B, C, D 上，即：

$$A = A_0 + A, B = B_0 + B, C = C_0 + C, D = D_0 + D，$$

（4）对剩下的信息块继续（1）——（3）的操作，直到最后一块。

（5）最后输出 128 比特散列值：$MD5(x) = A \| B \| C \| D$。

3．MD5 的主循环及主要操作：

结构如图 6-5 所示：

MD5 的轮主要操作为：

$$a = b + \left(\left(a + f(b,c,d) + M + t\right) <<< s\right)。$$

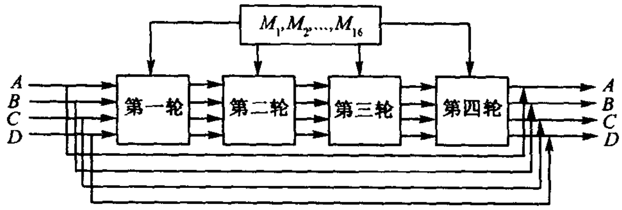


图 6-5　MD5 主循环

对应于四轮操作，f 分别取 F,G,H,I；对每一轮的 16 次运算，M 分别取 $M_1,M_2,\cdots,M_{16}$；

主要轮操作如图 6-6 所示：

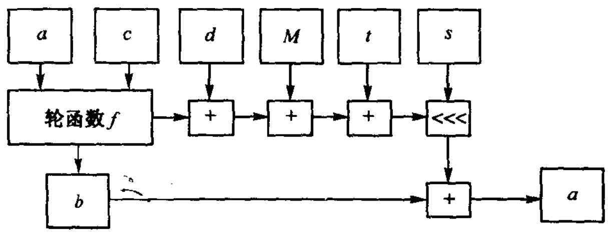


图 6-6　轮主要操作

6.3.2　SHA-1

1995 年，美国国家标准与技术研究所在 FIPS PUBS（Federal Information Processing Standards Publication 1995 April 17）上公布了新的散列函数标准，即 FIPS PUBS 180-1-1995，替代了 1993 年颁布的散列函数标准 PUBS 180-1-1993。新颁布的散列函数标准算法称为安全散列函数算法 SHA-1，使用范围为计算机安全，应用于文件或数据的压缩表示。

SHA-1 接受长度小于 2^{64} 比特的输入数据，输出 160 比特的散列函数值。SHA-1 是与数字签名标准 DSS 配套的散列函数标准，也就是说 SHA-1 的输出是 DSS 的输入。

SHA-1 是安全的，是一个强无碰撞散列函数，对于输入的任何微小的改变都会引起输出的很大差异。

1．基本操作和元素

（1）逐比特逻辑运算

或运算 $\vee$，与运算 $\wedge$，异或运算 $\oplus$，取反运算～和 MD5 相同；

（2）加法运算

令字长 w=32，若 $X \equiv x \bmod 2^w$，$Y \equiv y \bmod 2^w$，定义：

$$Z \equiv X + Y \bmod 2^w。$$

（3）移位操作

$x << n$ 表示向左移 n 个比特，右边空出的 n 个比特用 0 填充；

$x >> n$ 表示向右移 n 个比特，左边空出的 n 个比特用 0 填充。

设 X 是一个字，$0 \leqslant n < w$ 是一个整数，定义：

$X <<< n = (X << n) \vee (X >> w-n)$，即：<<<是 w 位循环左移位操作。

（4）基本函数组

SHA-1使用了3类，80个函数，这些函数作用在3个 w 比特的字上：

$$f_t(a,b,c)=\begin{cases}(a\wedge b)\vee(\tilde{b}\wedge c), & 0\leqslant t\leqslant 19\\ a\oplus b\oplus c, & 20\leqslant t\leqslant 39\\ (a\wedge b)\vee(a\wedge c)\vee(b\wedge c), & 40\leqslant t\leqslant 59\\ a\oplus b\oplus c, & 60\leqslant t\leqslant 79\end{cases}$$

（5）使用的常数

$$K_t=\begin{cases}\text{5A827999}, & 0\leqslant t\leqslant 19\\ \text{6ED9EBA1}, & 20\leqslant t\leqslant 39\\ \text{8F1BBCDC} & 40\leqslant t\leqslant 59\\ \text{CA62C1D6} & 60\leqslant t\leqslant 79\end{cases}$$

（6）消息分割和填充：SHA-1 的消息分组和填充同 MD5。

设信息被分割填充成 $x_1,x_2,\cdots,x_n$，每一个 x_i 被分成 16 个字 $M_0,M_1,\cdots,M_{15}$，M_0 是最左边的字。下面考虑 x_i 的处理过程。

2. 计算散列值

（1）初始化

$H_0^{(0)}=67452301$，$H_1^{(0)}=\text{EFCDAB89}$，$H_2^{(0)}=\text{98BADCFE}$，$H_3^{(0)}=10325476$，$H_4^{(0)}=$ C3D2E1F0；

（2）$W_t=\begin{cases}M_t, & 0\leqslant t\leqslant 15\\ (W_{t-3}\oplus W_{t-8}\oplus W_{t-14}\oplus W_{t-16})<<<1, & 16\leqslant t\leqslant 79\end{cases}$；

（3）令 $a=H_0^{(i-1)},b=H_1^{(i-1)},c=H_2^{(i-1)},d=H_3^{(i-1)},e=H_4^{(i-1)}$；

（4）对 $t=0\sim 79$：

$$Temp=a<<<5+f_t(b,c,d)+e+W_t+K_t,$$

$$e=d,\ d=c,\ c=b<<<30,\ b=a,\ a=Temp;$$

（5）$H_0^{(i)}=H_0^{(i-1)}+a$，$H_1^{(i)}=H_1^{(i-1)}+b$，$H_2^{(i)}=H_2^{(i-1)}+c$，$H_3^{(i)}=H_3^{(i-1)}+d$，$H_4^{(i)}=H_4^{(i-1)}+e$；

（6）以 $H_0^{(i)},H_1^{(i)},H_2^{(i)},H_3^{(i)},H_4^{(i)}$ 作为初始化常数接着对 x_{i+1} 处理，最后输出 16 比特散列值：

$$SHA_1(x)=H_0^{(n)}\,\|\,H_1^{(n)}\,\|\,H_2^{(n)}\,\|\,H_3^{(n)}\,\|\,H_4^{(n)}。$$

SHA-1 主要过程结构如图 6-7 所示。

3. 示例

（1）设原始信息为“abc”，其比特串为：01100001 01100010 01100011。它的长度 $l=24$，于是添加 1 个“1”，423 个“0”，再加上 64 比特串 $(24)_{10}$=00000000 00000018，即：

$$x=\overbrace{01100001}^{\text{"a"=32bits}}\ \overbrace{01100010}^{\text{"b"=32bits}}\ \overbrace{01100011}^{\text{"c"=32bits}}\ \overbrace{1}^{\text{界符}}\ \overbrace{0\cdots0}^{\text{423 bits}}\ \overbrace{00000000\ \ 00000018}^{\text{64 bits}}$$，共 512 比特，只有

一个分组。

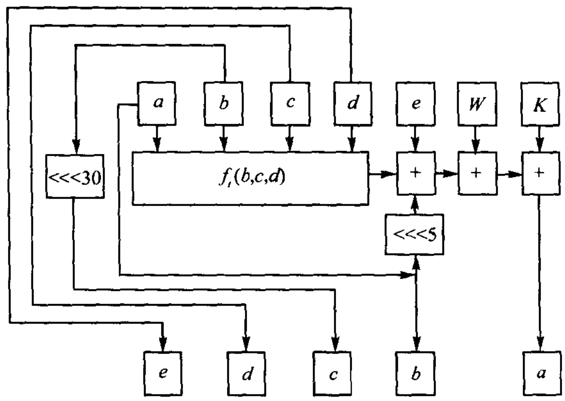


图 6-7　SHA-1 主要操作

（2）初始化散列值：

$$H_0 = 67452301,\ H_1 = \mathrm{EFCDAB89},\ H_2 = \mathrm{98BADCFE},$$
$$H_3 = 10325476,\ H_4 = \mathrm{C3D2E1F0};$$

（3）迭代计算：

W[0] = 61626380，W[1] =W[2] = …=W[14] =00000000，W[15] = 00000018；对 $t = 0$ 到 79 计算得到表 6-1。

表 6-1　迭代计算

	A	B	C	D	E
t=0	0116FC33	67452301	7BF36AE2	98BADCFE	10325476
t=1	8990536D	0116FC33	59D148C0	7BF36AE2	98BADCFE
t=2	A1390F08	8990536D	C045BF0C	59D148C0	7BF36AE2
t=3	CDD8E11B	A1390F08	626414DB	C045BF0C	59D148C0
t=4	CFD499DE	CDD8E11B	284E43C2	626414DB	C045BF0C
t=5	3FC7CA40	CFD499DE	F3763846	284E43C2	626414DB
t=6	993E30C1	3FC7CA40	B3F52677	F3763846	284E43C2
t=7	9E8C07D4	993E30C1	0FF1F290	B3F52677	F3763846
t=8	4B6AE328	9E8C07D4	664F8C30	0FF1F290	B3F52677
t=9	8351F929	4B6AE328	27A301F5	664F8C30	0FF1F290
t=10	FBDA9E89	8351F929	12DAB8CA	27A301F5	664F8C30
t=11	63188FE4	FBDA9E89	60D47E4A	12DAB8CA	27A301F5
t=12	4607B664	63188FE4	7EF6A7A2	60D47E4A	12DAB8CA
t=13	9128F695	4607B664	18C623F9	7EF6A7A2	60D47E4A
t=14	196BEE77	9128F695	1181ED99	18C623F9	7EF6A7A2
t=15	20BDD62F	196BEE77	644A3DA5	1181ED99	18C623F9
t=16	4E925823	20BDD62F	C65AFB9D	644A3DA5	1181ED99
t=17	82AA6728	4E925823	C82F758B	C65AFB9D	644A3DA5

（续）

	A	B	C	D	E
t=18	DC64901D	82AA6728	D3A49608	C82F758B	C65AFB9D
t=19	FD9E1D7D	DC64901D	20AA99CA	D3A49608	C82F758B
t=20	1A37B0CA	FD9E1D7D	77192407	20AA99CA	D3A49608
t=21	33A23BFC	1A37B0CA	7F67875F	77192407	20AA99CA
t=22	21283486	33A23BFC	868DEC32	7F67875F	77192407
t=23	D541F12D	21283486	0CE88EFF	868DEC32	7F67875F
t=24	C7567DC6	D541F12D	884A0D21	0CE88EFF	868DEC32
t=25	48413BA4	C7567DC6	75507C4B	884A0D21	0CE88EFF
t=26	BE35FBD5	48413BA4	B1D59F71	75507C4B	884A0D21
t=27	4AA84D97	BE35FBD5	12104EE9	B1D59F71	75507C4B
t=28	8370B52E	4AA84D97	6F8D7EF5	12104EE9	B1D59F71
t=29	C5FBAF5D	8370B52E	D2AA1365	6F8D7EF5	12104EE9
t=30	1267B407	C5FBAF5D	A0DC2D4B	D2AA1365	6F8D7EF5
t=31	3B845D33	1267B407	717EEBD7	A0DC2D4B	D2AA1365
t=32	046FAA0A	3B845D33	C499ED01	717EEBD7	A0DC2D4B
t=33	2C0EBC11	046FAA0A	CEE1174C	C499ED01	717EEBD7
t=34	21796AD4	2C0EBC11	811BEA82	CEE1174C	C499ED01
t=35	DCBBB0CB	21796AD4	4B03AF04	811BEA82	CEE1174C
t=36	0F511FD8	DCBBB0CB	085E5AB5	4B03AF04	811BEA82
t=37	DC63973F	0F511FD8	F72EEC32	085E5AB5	4B03AF04
t=38	4C986405	DC63973F	03D447F6	F72EEC32	085E5AB5
t=39	32DE1CBA	4C986405	F718E5CF	03D447F6	F72EEC32
t=40	FC87DEDF	32DE1CBA	53261901	F718E5CF	03D447F6
t=41	970A0D5C	FC87DEDF	8CB7872E	53261901	F718E5CF
t=42	7F193DC5	970A0D5C	FF21F7B7	8CB7872E	53261901
t=43	EE1B1AAF	7F193DC5	25C28357	FF21F7B7	8CB7872E
t=44	40F28E09	EE1B1AAF	5FC64F71	25C28357	FF21F7B7
t=45	1C51E1F2	40F28E09	FB86C6AB	5FC64F71	25C28357
t=46	A01B846C	1C51E1F2	503CA382	FB86C6AB	5FC64F71
t=47	BEAD02CA	A01B846C	8714787C	503CA382	FB86C6AB
t=48	BAF39337	BEAD02CA	2806E11B	8714787C	503CA382
t=49	120731C5	BAF39337	AFAB40B2	2806E11B	8714787C
t=50	641DB2CE	120731C5	EEBCE4CD	AFAB40B2	2806E11B
t=51	3847AD66	641DB2CE	4481CC71	EEBCE4CD	AFAB40B2
t=52	E490436D	3847AD66	99076CB3	4481CC71	EEBCE4CD
t=53	27E9F1D8	E490436D	8E11EB59	99076CB3	4481CC71
t=54	7B71F76D	27E9F1D8	792410DB	8E11EB59	99076CB3
t=55	5E6456AF	7B71F76D	09FA7C76	792410DB	8E11EB59
t=56	C846093F	5E6456AF	5EDC7DDB	09FA7C76	792410DB
t=57	D262FF50	C846093F	D79915AB	5EDC7DDB	09FA7C76

（续）

	A	B	C	D	E
t=58	09D785FD	D262FF50	F211824F	D79915AB	5EDC7DDB
t=59	3F52DE5A	09D785FD	3498BFD4	F211824F	D79915AB
t=60	D756C147	3F52DE5A	4275E17F	3498BFD4	F211824F
t=61	548C9CB2	D756C147	8FD4B796	4275E17F	3498BFD4
t=62	B66C020B	548C9CB2	F5D5B051	8FD4B796	4275E17F
t=63	6B61C9E1	B66C020B	9523272C	F5D5B051	8FD4B796
t=64	19DFA7AC	6B61C9E1	ED9B0082	9523272C	F5D5B051
t=65	101655F9	19DFA7AC	5AD87278	ED9B0082	9523272C
t=66	0C3DF2B4	101655F9	0677E9EB	5AD87278	ED9B0082
t=67	78DD4D2B	0C3DF2B4	4405957E	0677E9EB	5AD87278
t=68	497093C0	78DD4D2B	030F7CAD	4405957E	0677E9EB
t=69	3F2588C2	497093C0	DE37534A	030F7CAD	4405957E
t=70	C199F8C7	3F2588C2	125C24F0	DE37534A	030F7CAD
t=71	39859DE7	C199F8C7	8FC96230	125C24F0	DE37534A
t=72	EDB42DE4	39859DE7	F0667E31	8FC96230	125C24F0
t=73	11793F6F	EDB42DE4	CE616779	F0667E31	8FC96230
t=74	5EE76897	11793F6F	3B6D0B79	CE616779	F0667E31
t=75	63F7DAB7	5EE76897	C45E4FDB	3B6D0B79	CE616779
t=76	A079B7D9	63F7DAB7	D7B9DA25	C45E4FDB	3B6D0B79
t=77	860D21CC	A079B7D9	D8FDF6AD	D7B9DA25	C45E4FDB
t=78	5738D5E1	860D21CC	681E6DF6	D8FDF6AD	D7B9DA25
t=79	42541B35	5738D5E1	21834873	681E6DF6	D8FDF6AD

最后的输出为：

$$H_0=67452301+42541B35=A9993E36,$$
$$H_1=EFCDAB89+5738D5E1=4706816A,$$
$$H_2=98BADCFE+21834873=BA3E2571,$$
$$H_3=10325476+681E6DF6=7850C26C,$$
$$H_4=C3D2E1F0+D8FDF6AD=9CD0D89D。$$

于是：SHA_1（“ABC”）= A9993E36 4706816A BA3E2571 7850C26C 9CD0D89D。

6.3.3 SHA-256，SHA-384，SHA-512

SHA 系列散列函数还有：SHA-256，SHA-384，SHA-512，它们之间有很大的相似性。

SHA-1，SHA-256 的消息分组都是 512 比特，SHA-384，SHA-512 的消息分组是 1024 比特。

SHA-1，SHA-256 的消息分组、填充格式为：

$$*\cdots*\ 1\ 0\cdots0\ \overbrace{\text{length}(x)}^{64\ bits},$$

SHA-384，SHA-512 的消息分组、填充格式为：

$$*\cdots*\ 1\ 0\cdots0\ \overbrace{\text{length}(x)}^{128\text{ bits}}$$。

1. SHA-256

（1）64 个初始常数：$K_0^{(256)}, K_1^{(256)}, \cdots, K_{63}^{(256)}$ 为

```
428a2f98 71374491 b5c0fbcf e9b5dba5 3956c25b 59f111f1 923f82a4 ab1c5ed5
d807aa98 12835b01 243185be 550c7dc3 72be5d74 80deb1fe 9bdc06a7 c19bf174
e49b69c1 efbe4786 0fc19dc6 240ca1cc 2de92c6f 4a7484aa 5cb0a9dc 76f988da
983e5152 a831c66d b00327c8 bf597fc7 c6e00bf3 d5a79147 06ca6351 14292967
27b70a85 2e1b2138 4d2c6dfc 53380d13 650a7354 766a0abb 81c2c92e 92722c85
a2bfe8a1 a81a664b c24b8b70 c76c51a3 d192e819 d6990624 f40e3585 106aa070
19a4c116 1e376c08 2748774c 34b0bcb5 391c0cb3 4ed8aa4a 5b9cca4f 682e6ff3
748f82ee 78a5636f 84c87814 8cc70208 90befffa a4506ceb bef9a3f7 c67178f2
```

（2）8 个初始散列值 $H_0^{(0)}, \cdots, H_7^{(0)}$ 为

```
6a09e667 bb67ae85 3c6ef372 a54ff53a
510e527f 9b05688c 1f83d9ab 5be0cd19
```

（3）6 个基本函数为

$$Ch(x,y,z)=(x\wedge y)\oplus(\tilde{x}\wedge z)$$
$$Maj(x,y,z)=(x\wedge y)\oplus(x\wedge z)\oplus(y\wedge z)$$
$$\sum\nolimits_0^{(256)}(x)=(x<<<2)\oplus(x<<<13)\oplus(x<<<22)$$
$$\sum\nolimits_1^{(256)}(x)=(x<<<6)\oplus(x<<<11)\oplus(x<<<25)$$
$$\sigma_0^{(256)}=(x<<<7)\oplus(x<<<18)\oplus(x>>3)$$
$$\sigma_1^{(256)}=(x<<<17)\oplus(x<<<19)\oplus(x>>10)$$

其中 x,y,z 都是 32 比特。

（4）迭代过程

记：$X+Y:\equiv X+Y \bmod 2^{32}$，对所有消息块作

1）分割消息块 $x_i = M_0^{(i)}M_1^{(i)}\cdots M_{15}^{(i)}$；

2）$W_t=\begin{cases} M_t^{(i)}, & 0\leqslant t\leqslant 15 \\ \sigma_1^{(256)}(W_{t-2})+W_{t-7}+\sigma_0^{(256)}(W_{t-15})+W_{t-16}, & 16\leqslant t\leqslant 63 \end{cases}$；

3）令（$a,b,\cdots,h$）=（$H_0^{(i-1)},H_1^{(i-1)},\cdots,H_7^{(i-1)}$）；

4）对 $t=0,\cdots,63$ 作

$$T_1=h+\sum\nolimits_1^{(256)}(e)+Ch(e,f,g)+K_t^{(256)}+W_t$$
$$T_2=\sum\nolimits_0^{(256)}(a)+Maj(a,b,c)$$
$$h=g,g=f,f=e,e=d+T_1,d=c,c=b,b=a,a=T_1+T_2；$$

5）输出

$$H_0^{(i)}=a+H_0^{(i-1)},H_1^{(i)}=b+H_1^{(i-1)},H_2^{(i)}=c+H_2^{(i-1)},H_3^{(i)}=d+H_3^{(i-1)},$$

$$H_4^{(i)}=e+H_4^{(i-1)},H_5^{(i)}=f+H_5^{(i-1)},H_6^{(i)}=g+H_6^{(i-1)},H_7^{(i)}=h+H_7^{(i-1)}。$$

（5）最后输出散列值：$SHA_{256}(x)=H_0^{(n)}\parallel H_1^{(n)}\parallel\cdots\parallel H_7^{(n)}$。

2．SHA-384，SHA-512

（1）SHA-384，SHA-512 具有相同的 80 个常数：$K_0^{(512)},K_1^{(512)},\cdots,K_{79}^{(512)}$

```
428a2f98d728ae22 7137449123ef65cd b5c0fbcfec4d3b2f e9b5dba58189dbbc
3956c25bf348b538 59f111f1b605d019 923f82a4af194f9b ab1c5ed5da6d8118
d807aa98a3030242 12835b0145706fbe 243185be4ee4b28c 550c7dc3d5ffb4e2
72be5d74f27b896f 80deb1fe3b1696b1 9bdc06a725c71235 c19bf174cf692694
e49b69c19ef14ad2 efbe4786384f25e3 0fc19dc68b8cd5b5 240ca1cc77ac9c65
2de92c6f592b0275 4a7484aa6ea6e483 5cb0a9dcbd41fbd4 76f988da831153b5
983e5152ee66dfab a831c66d2db43210 b00327c898fb213f bf597fc7beef0ee4
c6e00bf33da88fc2 d5a79147930aa725 06ca6351e003826f 142929670a0e6e70
27b70a8546d22ffc 2e1b21385c26c926 4d2c6dfc5ac42aed 53380d139d95b3df
650a73548baf63de 766a0abb3c77b2a8 81c2c92e47edaee6 92722c851482353b
a2bfe8a14cf10364 a81a664bbc423001 c24b8b70d0f89791 c76c51a30654be30
d192e819d6ef5218 d69906245565a910 f40e35855771202a 106aa07032bbd1b8
19a4c116b8d2d0c8 1e376c085141ab53 2748774cdf8eeb99 34b0bcb5e19b48a8
391c0cb3c5c95a63 4ed8aa4ae3418acb 5b9cca4f7763e373 682e6ff3d6b2b8a3
748f82ee5defb2fc 78a5636f43172f60 84c87814a1f0ab72 8cc702081a6439ec
90befffa23631e28 a4506cebde82bde9 bef9a3f7b2c67915 c67178f2e372532b
ca273eceea26619c d186b8c721c0c207 eada7dd6cde0eb1e f57d4f7fee6ed178
06f067aa72176fba 0a637dc5a2c898a6 113f9804bef90dae 1b710b35131c471b
28db77f523047d84 32caab7b40c72493 3c9ebe0a15c9bebc 431d67c49c100d4c
4cc5d4becb3e42b6 597f299cfc657e2a 5fcb6fab3ad6faec 6c44198c4a475817
```

（2）初始散列值 $H_0^{(0)},\cdots,H_7^{(0)}$

SHA-384 的 8 个初始散列值为

cbbb9d5dc1059ed8，629a292a367cd507，9159015a3070dd17，152fecd8f70e5939，67332667ffc00b31，8eb44a8768581511，db0c2e0d64f98fa7，47b5481dbefa4fa4。

SHA-512 的 6 个初始散列值为

6a09e667f3bcc908，bb67ae8584caa73b，3c6ef372fe94f82b，a54ff53a5f1d36f1，510e527fade682d1，9b05688c2b3e6c1f，1f83d9abfb41bd6b，5be0cd19137e2179。

（3）SHA-384，SHA-512 具有相同的 6 个基本函数

$$Ch(x,y,z)=(x\wedge y)\oplus(\tilde{x}\wedge z)$$

$$Maj(x,y,z)=(x\wedge y)\oplus(x\wedge z)\oplus(y\wedge z)$$

$$\sum\nolimits_0^{\binom{384}{512}}(x)=(x<<<28)\oplus(x<<<34)\oplus(x<<<34)$$

$$\sum_1^{\binom{384}{512}}(x)=(x<<<14)\oplus(x<<<18)\oplus(x<<<41)$$

$$\sigma_0^{\binom{384}{512}}=(x<<<1)\oplus(x<<<8)\oplus(x>>7)$$

$$\sigma_1^{\binom{384}{512}}=(x<<<19)\oplus(x<<<61)\oplus(x>>6)$$

其中 x,y,z 都是 64 比特。

（4）SHA-384，SHA-512 具有相同的迭代算法。

记：$X+Y:\equiv X+Y \bmod 2^{64}$，对所有消息块作

1）分割消息块 $x_i=M_0^{(i)}M_1^{(i)}\cdots M_{15}^{(i)}$；

2）$W_t=\begin{cases} M_t^{(i)}, & 0\leqslant t\leqslant 15 \\ \sigma_1^{\binom{384}{512}}(W_{t-2})+W_{t-7}+\sigma_0^{\binom{384}{512}}(W_{t-15})+W_{t-16}, & 16\leqslant t\leqslant 79 \end{cases}$；

3）令（$a,b,\cdots,h$）$=\left(H_0^{(i-1)},H_1^{(i-1)},\cdots,H_7^{(i-1)}\right)$；

4）对 $t=0,\cdots,79$ 作

$$T_1=h+\sum_1^{\binom{384}{512}}(e)+Ch(e,f,g)+K_t^{\binom{384}{512}}+W_t$$

$$T_2=\sum_0^{\binom{384}{512}}(a)+Maj(a,b,c)$$

$$h=g,g=f,f=e,e=d+T_1,d=c,c=b,b=a,a=T_1+T_2;$$

5）输出

$$H_0^{(i)}=a+H_0^{(i-1)},H_1^{(i)}=b+H_1^{(i-1)},H_2^{(i)}=c+H_2^{(i-1)},H_3^{(i)}=d+H_3^{(i-1)},$$

$$H_5^{(t)}=f+H_5^{(i-1)},H_5^{(t)}=f+H_5^{(i-1)},H_6^{(i)}=g+H_6^{(i-1)},H_7^{(i)}=h+H_7^{(i-1)}。$$

（5）最后输出散列值

$$SHA_{384}(x)=H_0^{(n)}\parallel H_1^{(n)}\parallel\cdots\parallel H_4^{(n)}\parallel H_5^{(n)},$$

$$SHA_{512}(x)=H_0^{(n)}\parallel H_1^{(n)}\parallel\cdots\parallel H_4^{(n)}\parallel H_7^{(n)}。$$

6.3.4 MD5，SHA-1，SHA-256，SHA-384，SHA-512 的比较

MD5，SHA-1，SHA-256，SHA-384，SHA-512 这些散列函数的主要差别在于字长、初始常数、基本函数，它们的基本运算结构大体是相同的。

表 6-2 是它们的基本参数的比较。

表 6-2 MD5，SHA-1，SHA-256，SHA-384，SHA-512 性质比较表（长度单位为比特）

算法	消息长度	分组长	字长	输出长	初始常数	基本函数	操作步数
MD5	∞	512	32	128	4×32	4	4×16
SHA-1	$<2^{64}$	512	32	160	4×32	4	80
SHA-256	$<2^{64}$	512	32	256	64×32	6	64

（续）

算　法	消息长度	分　组　长	字　　长	输　出　长	初始常数	基本函数	操作步数
SHA-384	$<2^{128}$	1024	64	384	80×32	6	80
SHA-512	$<2^{128}$	1024	64	512	80×32	6	80

6.4 习题

1．设 Hash 函数为 $H(m)=m$，给出 Z_{89} 上的 ElGamal 签名方案。

2．设 ElGamal 签名中，Hash 函数为 $H(m)=m$，p=37859，α =24361，β=4046，秘密密钥为 e =1913。取随机数 t=2201，对消息 m=34201 进行签名。

3．设 ElGamal 签名中，Hash 函数为 $H(m)=m$，p=253109，α =332，β=223658，秘密密钥为 e =2835。取随机数 t =5151，对消息 m=2003 进行签名。

4．设 ElGamal 签名中，Hash 函数为 $H(m)=m$，p=29879，α =511，β=10419。已知消息 m=2002 的签名为（27765，10167）。假设随机数 t =119 泄露，求签名密钥 e。

5．设 ElGamal 签名中，Hash 函数为 $H(m)=m$，p=7949，α =123，β=6757。已知消息 m=114 的签名为（1676，1858）。假设随机数 t =119 泄露，求签名密钥 e。

6．设 ElGamal 签名中，Hash 函数为 $H(m)=m$，p=10531，α =3，β=9041。已知（9705，8376）是消息 m=2367 的签名，（9705，2780）是消息 m=8875 的签名，求随机数 t 和秘密密钥 e。

7．设 ElGamal 签名中，Hash 函数为 $H(m)=m$，p=37859，α =24361，β=26010。已知（37570，24926）是消息 m=9326 的签名，（37570，1780）是消息 m=10010 的签名，求随机数 t 和秘密密钥 e。

8．设：p=13759，q=2293，给出 DSS 签名方案。

9．设 DSS 中参数 p 为 20 比特素数 652411，q 为 10 比特素数 659。构造 DSS，取 Hash 函数为 $H(m)=m$，对消息 m=12345 签名。

10．设 DSS 中，Hash 函数为 $H(m)=m$，p=312863，q=14211，α =303316，β=26776，秘密密钥为 e =4321。对消息 m=123456 进行签名，并验证。

11．设 DSS 中，Hash 函数为 $H(m)=m$，p=356351，q=7127，α =313098，β=277711，秘密密钥为 e =1121。对消息 m=123456 进行签名，并验证。

12．设 D.Chaum 的 RSA 盲签名方案中，p=131，q=257，n=33667，e=71，d=7031。选择随机数 k=2672，对消息 m=35267 进行盲签名。

13．设 D.Chaum 的 RSA 盲签名方案中，p=239，q=379，n=90581，e=47，d=74651。选择随机数 k=3172，对消息 m=35267 进行盲签名。

14．设 Chaum－Antwerpen 不可否认签名方案中，p=37463，q=1873，e=971，d=1493，a=5929，b=9301。对消息 m=332 进行签名及不可否认验证。

15．设 Chaum－Antwerpen 不可否认签名方案中，p=57587，q=28793，e=747，d=11602，a=7744，b=26076。对消息 m=332 进行签名及不可否认验证。

16．设 p=57719，q=28859，α=2772，β=3014，是 Z_p^*的两个生成元，$h(x_1,x_2)=\alpha^{x_1}\beta^{x_2} \bmod p$ 为 Chaum–van Heijst–Pfitzmann 散列函数，(x_1,x_2)=(17279,14682)，(y_1,y_2)=(27869,7230)是一

个碰撞，计算离散对数 $\log_\alpha\beta$。

17．设 p=55103，q=27551，α=199，β=17808，是 Z_p^*的两个生成元，$h(x_1,x_2)=\alpha^{x_1}\beta^{x_2} \bmod p$ 为 Chaum–van Heijst–Pfitzmann 散列函数，(x_1,x_2)=(21702,1802)，(y_1,y_2)=(18920,5390)是一个碰撞，计算离散对数 $\log_\alpha\beta$。

18．证明 Merkle 级联算法构造的散列函数是强无碰撞的（提示，用反证法）。

19．证明：如果 f_1，f_2 是强无碰撞的，则 $f(x)=f_1(x)\|f_2(x)$定义的散列函数也是强无碰撞的。其中$\|$表示比特串的连接。

6.5 实验题

1．实现 MD5，检验其雪崩效应。

2．实现 SHA－1，检验其雪崩效应。

第7章 序列密码

7.1 序列密码模型

序列密码是密码技术的一个重要的分支。1949 年香农证明了只有一次一密密码体制是绝对安全的，给予序列密码技术的研究和应用强大的支持。序列密码也称为流密码。所谓流密码就是将密钥、明文表示成连续的符号或比特流，对应地进行加密，即：

设明文流为：$M = m_1 m_2 \cdots m_i \cdots$，

密钥流为：$K = k_1 k_2 \cdots k_i \cdots$，

则加密算法为：$C = c_1 c_2 \cdots c_i \cdots = E_{k_1}(m_1) E_{k_2}(m_2) \cdots E_{k_i}(m_i) \cdots$，

解密算法为：$M = m_1 m_2 \cdots m_i \cdots = D_{k_1}(c_1) D_{k_2}(c_2) \cdots D_{k_i}(c_i) \cdots$，

此模型如图 7-1 所示。

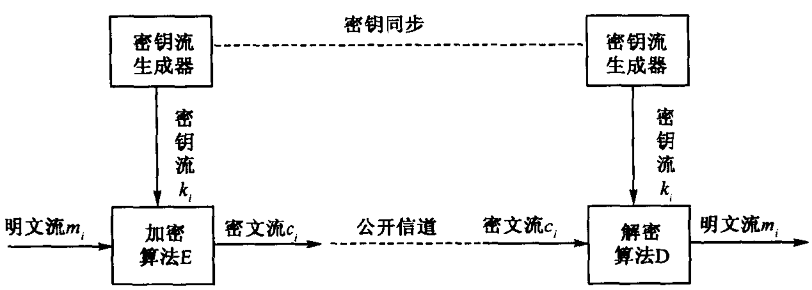


图 7-1 流密码保密通信模型

如果密钥流经过 l 个符号以后重复，那么这个流密码称为周期的，否则，称为非周期的。一次一密密码是非周期的，所谓一次一密密码体制描述如下：

设明文为比特流为：$M = m_1 m_2 \cdots m_i \cdots$；

密钥比特流为：$K = k_1 k_2 \cdots k_i \cdots$，则

加密算法为：$C = c_1 c_2 \cdots c_i \cdots = (m_1 \oplus k_1)(m_2 \oplus k_2) \cdots (m_i \oplus k_i) \cdots$，即：

$$c_i = m_i \oplus k_i \quad , \quad i = 1, 2, \cdots ;$$

解密算法和加密算法一样：

$C = c_1 c_2 \cdots c_i \cdots = (m_1 \oplus k_1)(m_2 \oplus k_2) \cdots (m_i \oplus k_i) \cdots$，即：

$$m_i = c_i \oplus k_i \quad , \quad i = 1, 2, \cdots ;$$

于是，其通信模型可简化为图 7-2。

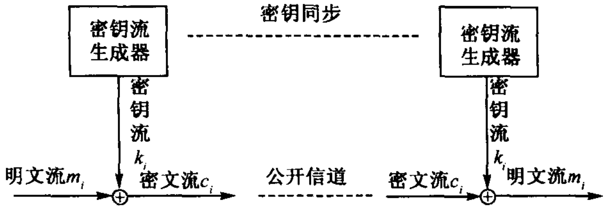


图 7-2　一次一密保密通信模型

可见，无论是一次一密密码体制还是一般的流密码体制，无限长密钥流是系统安全性的关键。这里有两个问题必须解决：

其一，密钥流的质量（随机性）如何刻画？如何保证？

其二，无限长密钥流如何产生？如何传递？加、解密端如何同步？

这就是流密码理论研究的主要内容。首先，我们来讨论第一个问题。

7.2　序列的随机性概念

在此仅考虑 0-1 序列，即密钥、明文流是由 0，1 构成的比特流。比特流的随机性直观上就是序列中 0，1 分布的随机性。数学上，所谓随机比特序列可以描述为：随机变量序列 $\xi_1\xi_2\cdots\xi_i\cdots$，其中 $\xi_i\ ,i=1,2,\cdots$ 是相互独立的，等概率取值 0，1 的随机变量。显然，这种随机性只有理论意义，密码应用中不可能产生这种绝对的随机序列。事实上，我们不但不可能产生真正的随机序列，也不可能产生无限长序列，也就是说，实际使用中的序列都是周期性的，但是，只要序列的周期足够大，比特的随机性足够好，就已满足密码技术应用的要求了。所以人们转而讨论一种所谓的伪随机性序列。下面逐步给出其定义。

定义 7-1　设有序列 $\{k_i \mid i=1,2\cdots\}$，如果存在正整数 P，使得对 $\forall i$，$k_{i+P}=k_i$，则 P 中最小者 p 称为序列的周期。

例如：0010111 0010111 0010111 0010111 0010111 ...... 有周期 7。

定义 7-2　在序列 $\{k_i \mid i=1,2\cdots\}$ 的一个周期 $k_{l+1},k_{l+2},\cdots,k_{l+p}$ 中，如果：$k_m \neq k_{m+1} = k_{m+2} = \cdots = k_{m+r} \neq k_{m+r+1}$，$l+1 \leqslant m+1 < m+r \leqslant l+p$，则 $(k_{m+1}k_{m+2}\cdots k_{m+r})$ 称为序列的一个 r-游程（run）。

如上例中，00 是序列的一个 0 的 2-游程，紧接着的 1 是一个 1 的 1-游程，再接着的 111 是一个 1 的 3-游程。

定义 7-3　设序列 $\{k_i \mid i=1,2\cdots\}$ 的周期为 p，令 $n_\tau = \#\{\ i \mid 1\leqslant i<p, k_i = k_{i+\tau}\}$，$d_\tau = \#\{\ i \mid 1\leqslant i<p, k_i \neq k_{i+\tau}\}$，则 $R(\tau)=\dfrac{n_\tau - d_\tau}{p}$，$\tau=0,1,2\cdots$ 称为序列 $\{k_i \mid i=1,2\cdots\}$ 的自相关函数。

显然，$R(np)=1$，$n=0,1,2,\cdots$，这称为同相自相关函数，其他情形称为异相自相关函数。

易见，$d_\tau = p - n_\tau$，所以，$R(\tau)=\dfrac{n_\tau-(p-n_\tau)}{p}=\dfrac{2n_\tau}{p}-1$，$n_\tau$ 就是延迟时间 τ 后的序列

与原序列在一个周期内相同比特的个数，反映了序列比特的均匀分布特性。如果 n_τ 是一个常数，则说明分布完全均匀，也就是说，通过这种平移比较得不到任何其他信息，这正是伪随机性的条件之一。在维吉利亚密码分析中正是使用这种平移比较（重合指数）来估计密钥长度的，可见这一条件是必要的。

1955 年，Golomb 提出序列伪随机性的三条假设，即 Golomb 随机性公设。

定义 7-4 设序列 $\{k_i \mid i=1,2\cdots\}$ 的周期为 p，考虑序列的一个周期，

（1）若 p 为偶数，则 0，1 的个数相等；若 p 是奇数，则 0，1 的个数相差 1；

（2）r-游程占总游程数的 $\frac{1}{2^r}$，$r=1,2,\cdots$；1 的 r-游程个数和 0 的 r-游程个数至多差 1；

（3）异相自相关函数 $R(\tau)$ 是一个常数。

满足以上三个条件的序列称为伪随机序列（Pseudo Random Sequence），也称为拟噪声序列（Pseudo Noise Sequence，PN 序列）。

伪随机序列应用于密码技术中还需要一些其他的条件，例如不可预测性，即根据部分比特不能够推测其他比特；周期要足够大，常常需要根据具体的序列生成器和应用来确定。

【例 7-1】 讨论序列：010011010001100111 010011010001100111 ......的随机性。

解 此序列的周期为 18，

0 的个数为 9，1 的个数也为 9；

0 的 1-游程个数为 2，1 的 1-游程个数也为 2；

0 的 2-游程个数为 2，1 的 2-游程个数也为 2；

0 的 3-游程个数为 1，1 的 3-游程个数也为 1；

游程总数为 10，

1-游程个数为 4，占总数的 $\frac{4}{10}=\frac{2}{5}\neq\frac{1}{2}$；

2-游程个数为 4，占总数的 $\frac{4}{10}=\frac{2}{5}\neq\frac{1}{2^2}$；

3-游程个数为 2，占总数的 $\frac{2}{10}=\frac{1}{5}\neq\frac{1}{2^3}$；

4-游程，5-游程，......，18 游程的个数都为 0，

$R(1)=\frac{2\times9}{18}-1=0$，$R(2)=\frac{2\times6}{18}-1=-\frac{1}{3}$，$R(3)=\frac{2\times8}{18}-1=-\frac{1}{9}$，$R(4)=\frac{2\times10}{18}-1=\frac{1}{9}$，
$R(5)=\frac{2\times8}{18}-1=-\frac{1}{9}$，$R(6)=\frac{2\times10}{18}-1=\frac{1}{9}$，$R(7)=\frac{2\times10}{18}-1=\frac{1}{9}$，$R(8)=\frac{2\times10}{18}-1=\frac{1}{9}$，
$R(9)=\frac{2\times4}{18}-1=-\frac{5}{9}$，$R(10)=\frac{2\times10}{18}-1=\frac{1}{9}$，$R(11)=\frac{2\times10}{18}-1=\frac{1}{9}$，$R(12)=\frac{2\times10}{18}-1=\frac{1}{9}$，
$R(13)=\frac{2\times8}{18}-1=-\frac{1}{9}$，$R(14)=\frac{2\times10}{18}-1=\frac{1}{9}$，$R(15)=\frac{2\times8}{18}-1=-\frac{1}{9}$，$R(16)=\frac{2\times6}{18}-1=-\frac{1}{3}$，
$R(17)=\frac{2\times8}{18}-1=-\frac{1}{9}$。

由此例可以熟悉游程和相关函数的计算方法。但本序列不是伪随机序列，因此也可以看到，伪随机序列的条件是很难严格满足的，常常只是近似地满足。本例可以说已经基本满足伪随机序列的条件。游程条件误差已不算大，相关函数条件可能吻合得更好。

【**例 7-2**】 讨论序列

1010111011000111110011010010000 1010111011000111110011010010000 ...的随机性。

解 此序列的周期为 31，0 的个数为 15，1 的个数为 6，

0 和 1 的 1-游程个数都为 4，0 和 1 的 2-游程个数都为 2，

0 和 1 的 3-游程个数都为 1，0 的 4-游程个数为 1，1 的 4-游程个数为 0，

0 的 5-游程个数为 0，1 的 5-游程个数为 1，$r>5$ 时，r-游程个数都为 0，

游程总数为 16，

1-游程数占总游程的 $\frac{8}{16}=\frac{1}{2}$，2-游程数占总游程的 $\frac{4}{16}=\frac{1}{2^2}$，3-游程数占总游程的 $\frac{2}{16}=\frac{1}{2^3}$，4-游程数占总游程的 $\frac{1}{16}=\frac{1}{2^4}$，5-游程数占总游程的 $\frac{1}{16}$，$r>5$ 时，r-游程总数占总游程的比例都为 0。

从此可以看出，Golomb 第二公设不可能绝对满足，特别当 r 比较大时，r-游程数占总游程的比例一般都为 0。所以 Golomb 第二公设应该修改为：$2^r\leqslant$游程总数时，r-游程的占总游程数的 $\frac{1}{2^r}$。

进一步容易验证：$\tau\neq 0$ 时，$R(\tau)=-\frac{1}{31}$。

以上结果说明，本序列是伪随机序列。

序列的随机性，还有其他的刻画方法，如：分布函数、不可预测性、线性复杂性等。

下面讨论 7.1 节提出的第二个问题。

7.3 线性反馈移位寄存器

寻找生成一个具有良好随机特性密码序列的方法是流密码技术和理论研究的重要工作。常见的有：线性移位寄存器 LFSR、非线性移位寄存器 NLFSR、有限自动机、线性同余等方法和新近提出的混沌密码序列技术。这些方法都是通过一个种子（有限长）产生具有足够长周期的、良好随机性的序列。在传递、存储序列时，只须传递、存储生成器的方法和种子。

线性反馈移位寄存器 LFSR（Linear Feedback Shift Register）是一种研究得非常成熟的序列生成方法，已被广泛地应用于密码技术、保密通信技术等方面。

定义 7-5 设 $f(x_1,x_2,\cdots,x_n)=c_1x_n+c_2x_{n-1}+\cdots+c_nx_1$ 是一个线性函数，c_i,x_i 是 0 或 1，$i=1,2,\cdots$，其中的加号表示异或。则对于给定的初始状态 $s_1=(a_1,a_2,\cdots,a_n)$，$a_{i+n+1}=f(a_{i+1},a_{i+2},\cdots,a_{i+n})=c_1a_{i+n}+c_2a_{i+n-1}+\cdots+c_na_{i+1},i=0,1,2,\cdots$ 称为一个 n 级线性反馈移位寄存器，f 称为反馈函数，$c_1,c_2,\cdots,c_n$ 称为反馈系数。

由此可以得到一个输出序列 $a_1,a_2,\cdots,a_{n-1},a_n,\cdots,a_i,\cdots$，和状态序列 $s_1,s_2,\cdots,s_{n-1},s_n,\cdots,s_i,\cdots$，$s_i=(a_i,a_{i+1},\cdots,a_{i+n-1})$，$i=0,1,\cdots$。

线性反馈移位寄存器如图 7-3 所示。

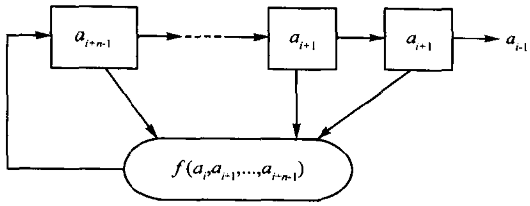


图 7-3　反馈移位寄存器

如果 $f(x_1,x_2,\cdots,x_n)=c_1x_n+c_2x_{n-1}+\cdots+c_nx_1$ 不是线性函数，则称该反馈移位寄存器是一个非线性反馈移位寄存器 NLFSR。这里对非线性移位寄存器序列不作进一步展开，仅以个别例子来说明。

【例 7-3】　设 $n=4, s_0=(1,0,1,0)$，$f(x_0,x_1,x_2,x_3)=x_0+x_1+x_2x_3$，计算其输出序列。

解　寄存器的输出为：1010 11010110101101011⋯。

下面考虑线性反馈移位寄存器的例子和性质。

【例 7-4】　设 $n=4, s_1=(1011), f(x_1,x_2,x_3,x_4)=x_1+x_4$，计算其输出序列。

解　寄存器如图 7-4 所示。

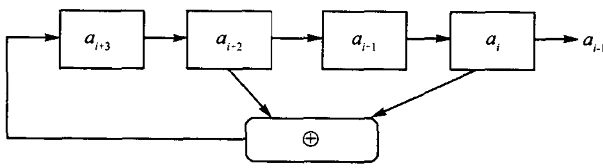


图 7-4　例 7-4 图

输出为：1011011011011011011011011011⋯。

可见，输出序列具有周期 3。

【例 7-5】　设 $n=4, s_1=(1011), f(x_1,x_2,x_3,x_4)=x_1+x_2$，计算其输出序列。

解　寄存器如图 7-5 所示。

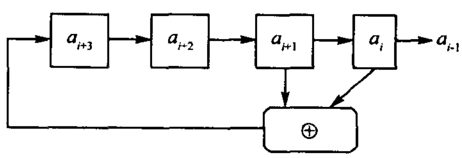


图 7-5　例 7-5 图

输出序列为：101111000100110 101111000100110 101111000100110 ....可见，序列具有周期 15。

从上两例可见，序列都是周期的，那么事实如何呢？

定理 7-1　n 级线性反馈移位寄存器输出序列是周期的，周期最大为 2^n-1。

证明　记状态序列为：$s_1,s_2,\cdots,s_i,\cdots$。

设 LFSR 的反馈系数为 $c_1,c_2,\cdots,c_n$，输出序列为 $a_1,a_2,\cdots,a_i,\cdots$，则：$a_{i+n}=c_1a_{i+n-1}+c_2a_{i+n-2}+\cdots+c_na_i, i=1,\cdots$。

考虑三个连续状态：$s_{i-1}=(a_{i-1},a_i,\cdots,a_{i+n-2})$，$s_i=(a_i,a_{i+1},\cdots,a_{i+n-1})$，$s_{i+1}=(a_{i+1},a_{i+2},\cdots,a_{i+n})$。

由于 $a_{i+n-1}=c_1a_{i+n-2}+c_2a_{i+n-3}+\cdots+c_na_{i-1}$，$c_n=1$，所以，$a_{i-1}=a_{i+n-1}-(c_1a_{i+n-2}+c_2a_{i+n-3}+\cdots+c_{n-1}a_i)$，可见已知 s_i 就能够求得 s_{i-1}。再由于 $a_{i+n}=c_1a_{i+n-1}+c_2a_{i+n-2}+\cdots+c_na_i$，所以已知 s_i 就能够求得 s_{i+1}，即：LFSR 的前一状态和后一状态都由当前状态完全确定。

由于 $s_1,s_2,\cdots,s_i,\cdots$，都是 0，1 的 n 元数组，所以最多有 2^n 个不同的情况，即必存在 $t\leqslant 2^n$，$s_1=s_t$。由于 s_1 完全确定 s_2，s_t 完全确定 s_{t+1}，所以必有 $s_2=s_{t+1}$，……，$s_{t+1}=s_{2t}$，……，从而 $s_i=s_{i+t}$，所以状态序列是周期的，周期不大于 2^n。

进一步，如果状态中有一个是 $(0,0,\cdots,0)$，那么其前、后所有的状态也都是 $(0,0,\cdots,0)$，所以此时的序列是一个全 0 序列，周期为 1。

从而，周期大于 1 的输出状态序列中，不同的状态最多有 2^n-1 个。即：LFSR 状态序列的周期最多为 2^n-1。

另一方面易见，输出序列的周期与输出状态序列一致，所以，LFSR 的输出序列的周期最多为 2^n-1。

7.4 m 序列及其随机性

定义 7-6 周期为 2^n-1 的 LFSR 序列称为 m 序列。

例 7-5 的序列是一个 m 序列。

m 序列的长周期是我们希望的。后面将证明 m 序列是一种伪随机序列。

m 序列必循环地遍历所有 2^n-1 个状态，不同的 m 序列只能是这些状态的不同排列。由此可见，如果 LFSR 的一个输出序列是 m 序列，则它的所有非 0 输出序列都是同一个 m 序列（可能前 2^n-2 个输出不同）。

那么，什么样的 LFSR 可以生成一个 m 序列呢？下面讨论这个问题。

定义 7-7 设 LFSR 的反馈系数为 $c_1,c_2,\cdots,c_n$，

即：$a_{i+n+1}=c_1a_{i+n}+c_2a_{i+n-1}+\cdots+c_na_{i+1},i=0,1,\cdots$，

则：$p_n(x)=1+c_1x+c_2x^2+\cdots+c_nx^n$ 称为 LFSR$\{a_i\mid i=0,1,\cdots\}$的特征多项式。易见 LFSR 由 $p_n(x)$完全确定，用 $G(p_n)$ 表示 $p_n(x)$的所有输出序列的集合，即，$G(p_n)=\{(a_1,a_2,\cdots,a_i\cdots)\mid a_{i+n+1}=c_1a_{i+n}+c_2a_{i+n-1}+\cdots+c_na_{i+1},i=0,1,\cdots\}$，则有下面的定理。

定理 7-2 $G(p_n)$ 是一个 n 维线性空间。

证明 设 $\{a_i\},\{b_i\}\in G(p_n)$，$s,t\in\{0,1\}=F_2$，$d_i=sa_i+tb_i$ ，$i=1,2,\cdots$，则：

$$\begin{aligned}
d_{i+n+1}&=sa_{i+n+1}+tb_{i+n+1}\\
&=s(c_1a_{i+n}+c_2a_{i+n-1}+\cdots+c_na_{i+1})+t(c_1b_{i+n}+c_2b_{i+n-1}+\cdots+c_nb_{i+1})\\
&=c_1(sa_{i+n}+tb_{i+n})+c_2(sa_{i+n-1}+tb_{i+n-1})+\cdots+c_n(sa_{i+1}+tb_{i+1})\\
&=c_1d_{i+n}+c_2d_{i+n-1}+\cdots+c_nd_{i+1}
\end{aligned}$$

于是，$\{d_i\}\in G(p_n)$，所以$G(p_n)$是一个线性空间。

另一方面，$\{a_i\}\neq\{b_i\}\Leftrightarrow(a_1,a_2,\cdots,a_n)\neq(b_1,b_2,\cdots,b_n)$，所以$G(p_n)$是 n 维的。

定理 7-3 如果$p_n(x)|q_m(x)$，则$G(p_n)$是$G(q_m)$的子空间，即：$G(p_m)\subset G(q_m)$。

证明 设$q_m(x)=p_n(x)r_k(x)$，

$$q_m(x)=c_0+c_1x+\cdots+c_mx^m,\ c_0=1,$$

$$p_n(x)=d_0+d_1x+\cdots+d_nx^n,\ d_0=1,$$

$$r_k(x)=e_0+e_1x+\cdots+e_kx^k,\ e_0=1,$$

则：$c_i=d_0e_i+d_1e_{i-1}+\cdots+d_{i-1}e_1+d_ie_0,\ i=1,2,\cdots,m,$

如果$\{a_i\}\in G(p_n)$，则有：

$$\begin{aligned}
a_{i+n+1}&=d_1a_{i+n}+d_2a_{i+n-1}+\cdots+d_na_{i+1},\\
a_{i+n+2}&=d_1a_{i+n+1}+d_2a_{i+n}+\cdots+d_na_{i+2},\\
&\vdots\\
a_{i+n+k}&=d_1a_{i+n+k-1}+d_2a_{i+n+k-2}+\cdots+d_na_{i+k}\\
a_{i+n+k+1}&=d_1a_{i+n+k}+d_2a_{i+n+k-1}+\cdots+d_na_{i+k+1}
\end{aligned},$$

分别用$e_k,e_{k-1},\cdots,e_2,e_1,e_0$乘以上各式并相加，得：

$$\begin{aligned}
a_{i+n+k+1}=&(d_1-e_1)a_{i+n+k}+(d_2+e_1d_1-e_2)a_{i+n+k-1}+\cdots\\
&+(d_j+e_1d_{j-1}+\cdots+e_{j-1}d_1-e_j)a_{i+n+k-j+1}+\cdots\\
&+(d_{n+k}+e_1d_{n+k-1}+\cdots+e_{n+k-1}d_1-e_{n+k})a_{i+1}
\end{aligned}$$

由于在二元异或运算下，加减相同，$m=n+k$，所以：

$$a_{i+m+1}=c_1a_{i+m}+c_2a_{i+m-1}+\cdots+c_ma_{i+1},$$

所以$\{a_i\}\in G(q_m)$，即$G(p_m)\subset G(q_m)$。

定义 7-8 设$p_n(x)$为 n 次多项式，则$l=\min\left\{k:\ p_n(x)\middle|(x^k+1)\right\}$称为多项式$p_n(x)$的阶。阶为$2^n-1$的不可约多项式称为本原多项式。

定理 7-4 设$p_n(x)$是一个 n 级 LFSR，$\{a_i\,|\,i=0,1,\cdots\}$是它的一个输出序列，如果$\{a_i\,|\,i=0,1,\cdots\}$具有周期2^n-1，则$p_n(x)$是不可约的。

证明 反设$q_n(x)=p_r(x)h_l(x)$，$0\leqslant r<n$，则由定理 7.4.1 知$G(p_r)\subset G(q_n)$。$G(p_r)$中序列的周期最多是$2^r-1<2^n-1$，但又知，$G(q_n)$中非 0 序列的周期都是2^n-1，于是矛盾。所以$p_n(x)$不可约。

但是，定理的逆不成立。

例如：经验证，多项式$x,x+1,x^2+x+1$不能整除$p_4(x)=x^4+x^3+x^2+x+1$，所以$p_4(x)$是不可约多项式。但是$p_4(x)$整除x^5+1，它的一个非 0 输出序列：000110001100011000110001100011...有周期 5，却没有周期$2^4-1=15$。

下面不加证明给出几个重要结论。

定理 7-5 非 0 序列$\{a_i\}\in G(p_n)$，$p_n(x)$是阶为 m 的不可约多项式，则$\{a_i\}$的周期为 m。

于是有

定理 7-6 $\{a_i\}\in G(p_n)$是 m 序列的充要条件是：$p_n(x)$是本原多项式。

可见，生成 m 序列的 n 阶 LFSR 的个数与 n 次本原多项式个数一致。于是，由多项式理论可得

定理 7-7 生成 m 序列的 n 阶 LFSR 的个数是$\dfrac{\varphi(2^n-1)}{n}$。

定理 7-8 设$\{a_i \mid i=1,2,\cdots\}$是一个周期序列，则存在惟一、首项系数为 1 的多项式 P，使得 $a_i\in G(P)$。

以上讨论了 LFSR 序列的一般性质，下面考虑 m 序列的伪随机性。

定理 7-9 m 序列是伪随机序列。

证明 设寄存器是 n 级的。

（1）第一公设

m 序列的输出状态必遍历2^n-1个状态：

$$\overbrace{100\cdots000}^{n},\ 100\cdots001,\ \cdots\cdots,111\cdots101,\ 111\cdots110,\ 111\cdots111,$$
$$000\cdots001,\ \cdots\cdots,011\cdots101,\ 011\cdots110,\ 011\cdots111;$$

即：除了全 0 状态以外的、n 个元的所有可能排列。

这些状态的首位恰恰是一个输出周期。而首位为 1 的排列共有2^{n-1}个，所以 m 序列的一个周期中有2^{n-1}个 1；首位为 0 的排列也有2^{n-1}个，但其中有一个全 0 状态，除去后，得到$2^{n-1}-1$个，所以 m 序列的一个周期中有$2^{n-1}-1$个 0；

（2）第二公设

1）由于寄存器不经过全 0 状态，所以 0 的$n-$游程的个数为 0；而寄存器必出现一个全 1 状态，所以 1 的$n-$游程的个数为 1。

2）1 的大于n的游程个数是 0，因否，会出现两个以上的全 1 状态，这是不可能的。0 的大于n的游程个数也是 0，因否，会出现全 0 状态。

3）1 的$n-$游程必呈如下状态：$0\underbrace{11\cdots11}_{n个}0$，由此可产生三个状态：$0\underbrace{1\cdots1}_{n-1个}$ $\underbrace{1\cdots1}_{n个}$ $\underbrace{1\cdots1}_{n-1个}0$，如果有一个 1 的$n-1-$游程，则必有$0\underbrace{1\cdots1}_{n-1个}0$，出现状态$0\underbrace{1\cdots1}_{n-1个}$，$\underbrace{1\cdots1}_{n-1个}0$，这是不可能的，因为每一个状态只能出现一次。所以 1 的$n-1-$游程个数为 0。

因为没有全 0 状态，所以，0 的 1-游程必呈$\underbrace{10\cdots01}_{n位}$状态，不会出现$\underbrace{10\cdots00}_{n位}$状态。于是得到一个 0 的$n-1-$游程$1\underbrace{0\cdots0}_{n-1个}1$。由此产生两个状态：$1\underbrace{0\cdots0}_{n-1个}$ $\underbrace{0\cdots0}_{n-1个}1$，如果再有其他 0 的$n-1-$游程$1\underbrace{0\cdots0}_{n-1个}1$，将也会产生这两个状态：$1\underbrace{0\cdots0}_{n-1个}$ $\underbrace{0\cdots0}_{n-1个}1$，由状态的惟一性可知，这是

不可能的，所以这个游程是惟一的，因此 0 的 $n-1-$游程的个数是 1。

4）如果 $n=2$，则 $n-1=1$，游程情况都已讨论。

于是设 $n>2$，$1\leqslant r<n-1$。1 的 $r-$游程为 $0\underbrace{1\cdots1}_{r}0$，它所在的状态为：$\underbrace{*\cdots*}_{s位}0\underbrace{1\cdots1}_{r位}0\underbrace{*\cdots*}_{t位}$，其中 $s+t=n-r-2$，所以 1 的 $r-$游程的个数为 2^{n-r-2}。类似地，0 的 $r-$游程的个数也为 2^{n-r-2}，$r-$游程数为 2^{n-r-1}。

5）0 与 1 的游程总数都为 $1+\sum_{r=1}^{n-2}2^{n-r-2}=2^{n-2}$，所以总游程数为 2^{n-1}。

一个周期中，0 的个数为 $2^{n-1}-1$个，1 的个数为 2^{n-1}个。

$r-$游程总数占总游程的 $\frac{2^{n-r-1}}{2^{n-1}}=\frac{1}{2^r}$，$1\leqslant r\leqslant n-1$，$n-$游程总数占总游程的 $\frac{1}{2^{n-1}}$，$r>n$ 时 $r-$游程总数为 0。

m 序列的游程见表 7-1。

表 7-1　m 序列的游程

$r-$游程	0 的游程	1 的游程	游　程	占总游程比
$1\leqslant r\leqslant n-2$	2^{n-r-2}	2^{n-r-2}	2^{n-r-1}	$1/2^r$
$r=n-1$	1	0	1	$1/2^r$
$r=n$	0	1	1	$1/2^{n-1}$
$r>n$	0	0	0	0

所以，m 序列满足第二公设。

（3）（换行定理）　设$\{a_i\}\in G(p_n)$是所讨论的 m 序列，$0<t<2^n-1$，令 $b_i=a_i+a_{i+t}$，可以验证，$\{b_i\}\in G(p_n)$，即$\{b_i\}$也是 m 序列，从而$\{b_i\}$的一个周期中 0 的个数为 $2^{n-1}-1$，此即$\{a_i\},\{a_{i+t}\}$一个周期中相同位的个数，从而 $R(t)=\frac{2^{n-1}-1-2^{n-1}}{2^n-1}=\frac{1}{1-2^n}$。所以 m 序列满足第三公设。

7.5　周期序列的线性复杂度

定义 7-9　生成周期序列的特征多项式的最小次数称为序列的线性复杂度。或者，设$\{a_i\}$是周期序列，则：

$L=\min\{l\mid a_{l+k+1}=c_1a_{l+k}+c_2a_{l+k-1}+\cdots+c_{l-1}a_{k+1}+c_la_k,k=1,2,\cdots\}$ 称为序列 $\{a_i\}$ 的线性复杂度。

即：序列的线性复杂度就是生成该序列的线性反馈移位寄存器的最小级数。

定理 7-10　设序列$\{a_i\}$的周期为 n，则序列的线性复杂度为循环矩阵

$$M(\{a_i\})=\begin{pmatrix} a_1 & a_2 & \cdots & a_n \\ a_n & a_1 & \cdots & a_{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_2 & a_3 & \cdots & a_1 \end{pmatrix}$$

的秩。

序列的线性复杂性是度量随机性的一项重要指标，在此不作展开，请参看有关序列密码的著作。

7.6 习题

1. 计算周期 19 的二元序列：

$\{a_i | i=1,2,\cdots,\}$=1010011101110101100 1010011101110101100 10100111011101…的游程数及各种游程所占的比例，自相关函数值。

2. 计算周期 20 的二元序列：

$\{a_i | i=1,2,\cdots,\}$=1101011101100111000111010111011001110001110101110110011100…的游程数及各种游程所占的比例，自相关函数值。

3. 设初始状态为 $a_5\,a_4\,a_3\,a_2\,a_1$=10011，计算 5 级二元反馈移位寄存器的周期，游程，自相关函数值：

$a_i(t+1)=a_{i+1}(t)$，$i=1,2,\cdots,4$，

$a_5(t+1)=a_1(t)\,a_2(t)+a_1(t)\,a_2(t)\,a_3(t)+a_1(t)\,a_3(t)+a_1(t)\,a_5(t)$。

4. 设初始状态为 $a_7\,a_6\,a_5\,a_4\,a_3\,a_2\,a_1$=1001101，计算 7 级二元反馈移位寄存器的周期，游程，自相关函数值，见图 7-6。

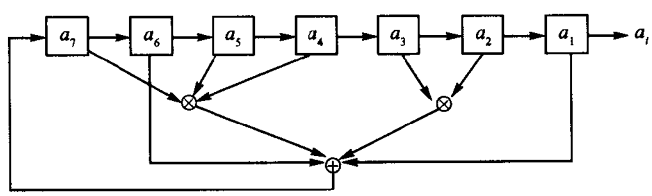


图 7-6 第 4 题图

其中⊗，⊕分别表示比特与，比特异或。

5. 设初始状态为 $a_6\,a_5\,a_4\,a_3\,a_2\,a_1$=111101，计算 6 级 LFRS 的周期，游程，自相关函数值：

$a_i(t+1)=a_{i+1}(t)$，$i=1,2,\cdots,5$，

$a_6(t+1)=a_1(t)+a_3(t)+a_5(t)+a_6(t)$。

6. 设初始状态为 $a_5\,a_4\,a_3\,a_2\,a_1$=10001，计算 6 级 LFRS 的周期，游程，自相关函数值，见图 7-7。

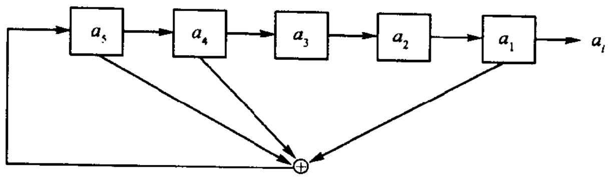


图 7-7　第 6 题图

7．设初始状态为 $a_6\,a_5\,a_4\,a_3\,a_2\,a_1$=101011，6 级 LFRS 的特征多项式为 $P_6(x)=1+x+x^3+x^5$，讨论 LFSR{ a_i | i=1,2,⋯} 的随机性。

8．构造一个 5 级 LFSR 的 m 序列。

第8章　椭圆曲线密码学

8.1　引言

椭圆曲线理论是一个古老而深奥的数学分支，已有100多年的历史，一直作为一门纯理论学科被少数数学家掌握。它被广大科技工作者了解要归功于20世纪80年代的两件重要的工作。第一，Weil 应用椭圆曲线理论证明了著名的费尔玛大定理。第二，Neal Koblitz 和 V.S.Miller 把椭圆曲线群引入公钥密码理论中，提出了基于椭圆曲线的公钥密码体制 ECC（Elliptic Curves Cryptosystem），取得了公钥密码理论和应用的突破性进展。

20世纪90年代，最通用的公钥密码体制是RSA公钥密码体制和Diffie-Hellman公钥密码交换算法。其密钥长度一般为512比特。1999年8月22日RSA-512被攻破，所以，这些公钥不得不被加长。为了达到对称密钥128比特的安全水平，NIST推荐使用3072比特的RSA密钥。显然这种密钥长度的增长，对本来计算速度缓慢的RSA来说，无疑是雪上加霜。ECC的提出改变了这种状况，实现了密钥效率的重大突破，大有以强大的短密钥优势取代RSA成为新一代公钥标准（事实标准）之势。

一般来说，ECC没有亚指数攻击，所以，它的密钥长度大大地减少，256比特的ECC密钥就可以达到对称密钥128比特的安全水平，表8-1是有关密码体制密钥长度的对照：

表8-1　ECC与Skipjack，3-DES，AES密钥长度对照表

ECC-p 密钥长度	ECC-2^m 密钥长度	其它算法及密钥长度	
$\|p\|=192$	m=163	Skipjack	Key=80
$\|p\|=224$	m=233	3-DES	Key=112
$\|p\|=256$	m=283	AES-small	Key=128
$\|p\|=384$	m=409	AES-Medium	Key=192
$\|p\|=521$	m=571	AES-Large	Key=256

可见，ECC密钥的优势不是其他公钥体制可比的。

ECC的安全性和优势得到了业界的认可和广泛的应用：

（1）1998年被确定为ISO/IEC数字签名标准ISO14888-3；

（2）1999年2月ECDSA（椭圆曲线数字签名算法）被ANSI确定为数字签名标准ANSI X9.62-1998，ECDH（椭圆曲线Diffie-Hellman体制版本）被确定为ANSI X9.63；

（3）2000年2月被确定为IEEE标准IEEE1363-2000，同期，NIST确定其为联邦数字签名标准FIPS186-2。

ECC还被确定为其它标准，如高效密码SEG等。这些标准并不完全独立，例如，ECDSA的各标准之间的包含关系如图8-1所示。

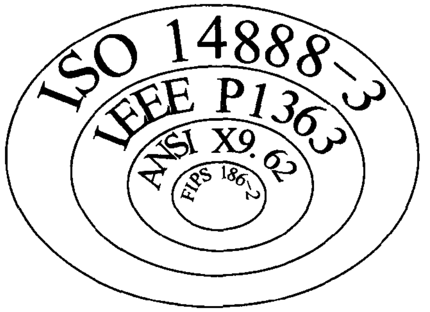


图 8-1　ECDSA 各标准之间的包含关系

8.2　椭圆曲线的概念

8.2.1　概念和定义

设 F_q 是一个元素个数为 q 的有限域，$q=p^r$，p 是一个素数，K 是一个域。令 $K^*=K\setminus\{0\}$，$\bar{K}$ 表示 K 的代数闭包。如果 $K=F_q$，则 $\bar{K}=\bigcup\limits_{r\geqslant 1}F_{q^r}$。

定义 8-1　对于任意的 $(x_1,y_1,z_1),(x_2,y_2,z_2)$，如果存在 $u\in K^*$，使得 $x_1=ux_2$，$y_1=uy_2$，$z_1=uz_2$，则称 $P^2(K)=K^3\setminus\{(0,0,0)\}/\sim$ 为 K 上的射影平面。$P^2(K)$ 中点其实是一个等价类，常记为 $(X:Y:Z)$。

定义 8-2　3 次齐次方程：

$$Y^2Z+a_1XYZ+a_3YZ^2=X^3+a_2X^2Z+a_4XZ^2+a_6Z^3 \tag{8-1}$$

称为 Weierstrass 方程，其中 $a_1,a_2,a_3,a_4,a_6\in\bar{K}$。

记 $F(X,Y,Z)=Y^2Z+a_1XYZ+a_3YZ^2-X^3-a_2X^2Z-a_4XZ^2-a_6Z^3$，$E(\bar{K})=\{(X,Y,Z)|(X,Y,Z)\in P^2(\bar{K}),F(X,Y,Z)=0\}$。

若 $P=(X,Y,Z)\in E(\bar{K})$，且 $\left(\frac{\partial F}{\partial X},\frac{\partial F}{\partial Y},\frac{\partial F}{\partial Z}\right)\neq 0$，则称点 P 是光滑的，或非奇异的。否则称 P 是一个奇异点，Weierstrass 方程称为奇异的。

定义 8-3　由光滑 Weierstrass 方程在 $P^2(\bar{K})$ 上的所有点构成的集合称为椭圆曲线（Elliptic Curve），通常记为 E。椭圆曲线上有一个必然存在的点 $(0:1:0)$，记为 O，称为无穷远点。

用等价类讨论问题太抽象，所以我们将椭圆曲线的齐次形式化为非齐次形式（称为仿射形式）。

设 $Z\neq 0$，令 $x=\frac{X}{Z},y=\frac{Y}{Z}$，则 Weierstrass 方程化为：

$$y^2+a_1xy+a_3y=x^3+a_2x^2+a_4x+a_6 \tag{8-2}$$

则椭圆曲线就是$A^2(\bar{K})=\bar{K}\times\bar{K}$中满足方程（8-2-2）的点集及无穷远点 O。

如果$a_1,a_2,a_3,a_4,a_6\in K$，则称椭圆曲线 E 定义在 K 上，记为 E/K。E/K 在$K^2=K\times K$中的点称为K－有理点，E/K 的全体K－有理点（包括无穷远点 O）记为 $E(K)$。

一般地考虑 E/K，简记为 E（含无穷远点 O）。

【例 8-1】 椭圆曲线 E：$y^2=x^3-36x^2$ 如图 8-2 所示：

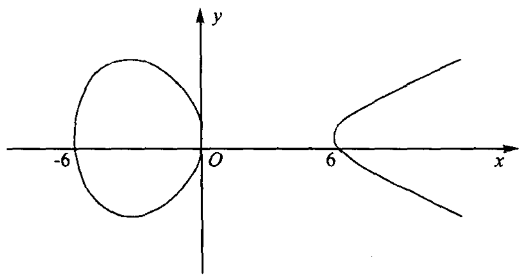


图 8-2 $y^2=x^3-36x$ 示意图

8.2.2 群运算

定义 8-4 设 E 是椭圆曲线，O 是无穷远点，定义：

（1）对$\forall P\in E$，$O+P=P+O=O$；

（2）$-O=O$；

（3）对$\forall P\in E$，$P=(x_1,y_1)\neq O$，令：$-P=(x_1,-y_1-a_1x_1-a_3)$；

（4）如果$Q=-P$，令$P+Q=O$；

（5）如果$P\neq O,Q\neq O,Q\neq -P,P=(x_1,y_1),Q=(x_2,y_2)$，定义：$P+Q=R=(x_3,y_3)$，定义：

$$\begin{cases}x_3=k^2+a_1k-a_2-x_1-x_2\\ y_3=-(k+a_1)x_3-l-a_3\end{cases},$$

$$k=\begin{cases}\dfrac{y_2-y_1}{x_2-x_1}, & P\neq Q\\ \dfrac{3x_1^2+2a_2x_1+a_4-a_1y_1}{2y_1+a_1x_1+a_3}, & P=Q\end{cases},\quad l=y_1-kx_1。$$

定理 8-1 对以上定义的运算，E 是一个 Abel 群，零元就是无穷远点 O。

以上运算的定义，可借助于几何方式来描述，以图 8-2 为参照。

设 P，Q 是 E 上的点，则：

（1）$-P$ 是过 P 的铅垂线与 E 的另一个交点，特别地，如果 P=（−6，0），（0，0），（0，6），则：$-P=P$，见图 8-3；

（2）设 l 是连接 P，Q 的直线，R 是 l 与 E 的第三个交点，则，$P+Q=-R$，特别地，如果 $P=Q$，则 l 成为切线，见图 8-4；

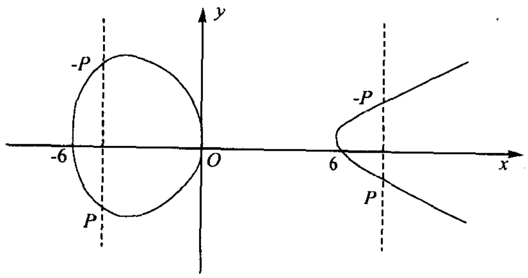


图 8-3 −P 示意图

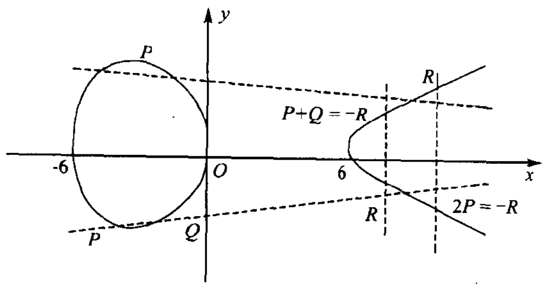


图 8-4 $P+Q$，$2P$ 示意图

【例 8-2】 设 $K=Q$ 为有理数域，E: $y^2=x^3-36x$，令 $P_1=(-3,9)$，$P_2=(12,36)$，$P_1,P_2\in E$，计算 P_1+P_2，$2P_1$。

解 $a_1=a_2=a_3=a_6=0$，$a_4=-36$。

（1）$k=\dfrac{y_2-y_1}{x_2-x_1}=\dfrac{36-9}{12+3}=\dfrac{9}{5}$，$l=y_1-kx_1=9-\dfrac{9}{5}\times(-3)=\dfrac{72}{5}$，

$$x_3=k^2+a_1k-a_2-x_1-x_2=\left(\frac{9}{5}\right)^2+3-12=-\frac{144}{25},$$

$$y_3=-(k+a_1)x_3-l-a_3=-\frac{9}{5}\times\left(-\frac{144}{25}\right)-\frac{72}{5}=-\frac{504}{125},$$

所以 $P_1+P_2=\left(-\dfrac{144}{25},-\dfrac{504}{125}\right)$。

（2）$k=\dfrac{3x_1^2+2a_2x_1+a_4-a_1y_1}{2y_1+a_1x_1+a_3}=\dfrac{3(-3)^2-36}{2\times9}=-\dfrac{1}{2}$，$l=y_1-kx_1=9-\dfrac{3}{2}=\dfrac{15}{2}$，

$x=\dfrac{25}{4}$，$y=-\dfrac{35}{8}$，所以 $2P=\left(\dfrac{25}{4},-\dfrac{35}{8}\right)$。

【例 8-3】 考虑 F_{23} 上的椭圆曲线 E：$y^2=x^3+x+1$。令 $P_1=(3,10)$，$P_2=(9,7)$，计算 P_1+P_2。

解 易验证：$E(Z_{23})=\{O,(0,1),(0,22),(1,7),(1,16),(3,10),(3,13),(4,0),(5,4)$，$(5,19),(6,4),$

(6,19),(7,12),(9,7),(9,16),(11,3),(11,20),(12,4),(12,19),(13,7),(13,16),(17,3),(17,20),(18,3),(18,20),(19,5),(19,18)}。

$$a_1 = a_2 = a_3 = 0\text{，}\quad a_4 = a_6 = 1\text{。}$$

（1）$k=\dfrac{y_2-y_1}{x_2-x_1}=\dfrac{7-10}{9-3}=-\dfrac{1}{2}=11$，$l=y_1-kx_1=10-11\times3=-23=0$，

$x_3=k^2+a_1k-a_2-x_1-x_2=11^2-3-9=109=17$，

$y_3=-(k+a_1)x_3-l-a_3=-11\times17=20$。

所以 $P_1+P_2=(17,20)$。

（2）$k=\dfrac{3x_1^2+2a_2x_1+a_4-a_1y_1}{2y_1+a_1x_1+a_3}=\dfrac{3\times3^2+1}{2\times10}=\dfrac{28}{20}=\dfrac{5}{20}=\dfrac{1}{4}=6$，$l=y_1-kx_1=10-6\times3=-8=15$，

$x=7$，$y=12$，所以 $2P=(7,12)$。

8.2.3 椭圆曲线的分类

定义 8-5 设 E_1,E_2 是定义在域 K 上的两个椭圆曲线，当 E_1,E_2 视为两个射影簇时，如果存在态射（Morphism）f: $E_1\to E_2$，g: $E_2\to E_1$，满足：

$g(f(P))=id_{E_1},f(g(S))=id_{E_2}$，其中 id_{E_1}，id_{E_2} 是 E_1,E_2 上的恒等映射，

则称 E_1,E_2 同构。关于射影簇态射的概念，可参阅有关代数几何的书籍。

定理 8-2 设椭圆曲线 E_1/K，E_2/K 由方程：

$$E_1:\ y^2+a_1xy+a_3y=x^3+a_2x^2+a_4x+a_6$$

$$E_2:\ y^2+\bar{a}_1xy+\bar{a}_3y=x^3+\bar{a}_2x^2+\bar{a}_4x+\bar{a}_6$$

给出，则 E_1/K，E_2/K 在 K 上同构的充要条件是：存在变量代换：

$$\pi:(x,y)\to\left(u^2x+r,u^3y+u^2sx+t\right) \tag{8-3}$$

$u,r,s,t\in K,u\neq0$，将 E_1 映到 E_2，（8-2-3）称为 E_1 到 E_2 的允许变量代换。

如果 E_1/K，E_2/K 同构，且 π 映 E_1 到 E_2，则变换：

$$\pi^{-1}:(x,y)\to\left(u^{-2}(x-r),u^{-3}(y-sx-t+rs)\right) \tag{8-4}$$

映 E_2 到 E_1，是 E_2 到 E_1 的允许变量代换。

如果 $E_1\overset{\pi}{\cong}E_2$，则：

$$\left.\begin{aligned}
u\bar{a}_1&=a_1+2s\\
u^2\bar{a}_2&=a_2-sa_1+3r-s^2\\
u^3\bar{a}_4&=a_4-sa_3+2ra_2-(t+rs)a_1+3r^2-2st\\
u^6\bar{a}_6&=a_6+ra_4+r^2a_2+r^3-ta_3-t^2-rta_1
\end{aligned}\right\} \tag{8-5}$$

于是有下面的

定理 8-3 E_1/K，E_2/K 同构的充要条件：存在 $u,r,s,t\in K,u\neq0$，使（8-5）成立。

定义 8-6 设 E 是定义在域 K 上的椭圆曲线，由（8-2）表示，令：

$$d_2=a_1^2+4a_2\text{，}\quad d_4=2a_4+a_1a_3\text{，}\quad d_6=a_3^2+4a_6$$

$d_8=a_1^2a_6+4a_2a_6-a_1a_3a_4+a_2a_3^2-a_4^2$，$c_4=d_2^2-24d_4$，定义：

$$\Delta=-d_2^2d_8-8d_4^3-27d_6^2+9d_2d_4d_6 \tag{8-6}$$

$$j(E)=c_4^3/\Delta,\quad \Delta\neq 0 \tag{8-7}$$

Δ 称为 Weierstrass 方程的判别式，$j(E)$ 称为 E 的 $j-$ 不变量。

定理 8-4 Weierstrass 方程表示的曲线 E 是椭圆曲线的充要条件是 $\Delta\neq 0$。

定理 8-5 若 $E_1/K\cong E_2/K$，则 $j(E_1)=j(E_2)$，反之，如果 K 是代数闭域，$j(E_1)=j(E_2)$，则：$E_1/K\cong E_2/K$。

设 E 定义在有限域 K 上，

（1）$Char(K)\neq 2$ 时：对（8-2）作变量代换

π：$(x,y)\rightarrow\left(x,y-\frac{a_1}{2}x-\frac{a_3}{2}\right)$，则（8-2）成为：

$$E'/K:\ y^2=x^3+b_2x^2+b_4x+b_6,\quad b_2,b_4,b_6\in K \tag{8-8}$$

且 $E/K\cong E'/K$。

$$\Delta(E')=-4\left(16b_4^3-72b_2b_4b_6+108b_2^2-4b_2^2\left(b_4^2-4b_2b_6\right)\right),$$

$$j(E')=-\frac{\left(4b_2^2-12b_4\right)^3}{16b_4^3-72b_2b_4b_6+108b_2^2-4b_2^2\left(b_4^2-4b_2b_6\right)}。$$

（2）$Char(K)\neq 2,3$ 时：进一步对（8-8）作变量代换

π：$(x,y)\rightarrow\left(\frac{x-3b_2}{36},\frac{y}{216}\right)$，则（8-8）成为：

$$E''/K:\ y^2=x^3+ax+b,\quad a,b\in K \tag{8-9}$$

有：$E/K\cong E'/K\cong E''/K$。

$$\Delta(E'')=-16\left(4a^3+27b^2\right),\quad j(E)=-1728(4a)^3/\Delta。$$

（3）$Char(K)=2$ 时：设椭圆曲线 E/K 的 Weierstrass 方程为：

$$E:\ y^2+\bar{a}_1xy+\bar{a}_3y=x^3+\bar{a}_2x^2+\bar{a}_4x+\bar{a}_6$$

则其 $j-$ 不变量 $j(E)=(\bar{a}_1)^{12}/\Delta$。

1）如果 $j(E)\neq 0$，则 $\bar{a}_1\neq 0$，那么有允许变量代换

$(x,y)\rightarrow\left(\bar{a}_1^2x+\frac{\bar{a}_3}{\bar{a}_1},\bar{a}_1^3y+\frac{\bar{a}_1^2\bar{a}_4+\bar{a}_3^2}{\bar{a}_1^3}\right)$，将 E 变换成：

$$E_1/K:\ y^2+xy=x^3+a_2x^2+a_6 \tag{8-10}$$

有：$E/K\cong E_1/K$，且 $\Delta(E_1)=a_6$，$j(E_1)=\frac{1}{a_6}$。

2）如果 $j(E)=0$，则 $\bar{a}_1=0$，那么有允许变量代换：

$(x,y)\rightarrow(x+\bar{a}_2,y)$，将 E 变换成：

$$E_2/K:\ y^2+a_3y=x^3+a_4x+a_6 \qquad (8\text{-}11)$$

有：$E/K \cong E_1/K$，且$\Delta(E_2)=a_3^4$，$j(E_2)=0$。

8.2.4 椭圆曲线分类加法公式

设$P=(x_1,y_1)$，$Q=(x_2,y_2)$，$R=(x_3,y_3)$，

（1）$Char(K)\neq 2,3$时：E/K：$y^2=x^3+ax+b$，a，$b\in K$。

1）$-P=(x_1,-y_1)$；

2）如果$Q\neq -P$，$R=P+Q$，那么：

$$\begin{cases} x_3=k^2-x_1-x_2 \\ y_3=k(x_1-x_3)-y_1 \end{cases}，且 k=\begin{cases} \dfrac{y_2-y_1}{x_2-x_1}, & P\neq Q \\ \dfrac{3x_1^3+a}{2y_1}, & P=Q \end{cases}$$

（2）$Char(K)=2$，$j(E)\neq 0$时：E/K：$y^2+xy=x^3+ax+b$，a，$b\in K$。

1）$-P=(x_1,y_1+x_1)$；

2）如果$Q\neq -P$，$R=P+Q$，那么：

$$x_3=\begin{cases} \left(\dfrac{y_1+y_2}{x_1+x_2}\right)^2+\dfrac{y_1+y_2}{x_1+x_2}+x_1+x_2+a, & P\neq Q \\ x_1^2+\dfrac{b}{x_1^2}, & P=Q \end{cases}$$

$$y_3=\begin{cases} \left(\dfrac{y_1+y_2}{x_1+x_2}\right)(x_1+x_3)+x_3+y_1, & P\neq Q \\ x_1^2+\left(x_1+\dfrac{y_1}{x_1}\right)x_3+x_3, & P=Q \end{cases}$$

（3）$Char(K)=2$，$j(E)=0$时：E/K：$y^2+ay=x^3+bx+c$，a，b，$c\in K$。

1）$-P=(x_1,y_1+a)$；

2）如果$Q\neq -P$，$R=P+Q$，那么：

$$x_3=\begin{cases} \left(\dfrac{y_1+y_2}{x_1+x_2}\right)^2+x_1+x_2, & P\neq Q \\ \dfrac{x_1^4+b^2}{a^2}, & P=Q \end{cases}$$

$$y_3=\begin{cases} \left(\dfrac{y_1+y_2}{x_1+x_2}\right)(x_1+x_3)+y_1+a, & P\neq Q \\ \left(\dfrac{x_1^2+b}{a}\right)(x_1+x_3)+y_1+a, & P=Q \end{cases}$$

8.3 椭圆曲线群的结构

8.3.1 群的阶

我们已经知道椭圆曲线上的点集可构成一个 Abel 群，而将要讨论的密码体制就是在这个群上进行的，所以，这个群的结构和阶对密码体制的安全性是至关重要的。研究这个群的结构和阶需要更加深刻、复杂的群理论基础。这里仅给出几个重要的定理和结论。

定理 8-6 （Hasse 定理）令 $\#E\left(F_q\right)=q+1-t$，则 $|t|\leqslant 2\sqrt{q}$。

定义 8-7 如果 $p|t$ 则称椭圆曲线是超奇异的（简称超奇的），否则称为非超奇的。

定理 8-7 （Weil 定理）设 E 是定义在有限域 F_q 上的椭圆曲线，令 $t=q+1-\#E\left(F_q\right)$，则 $\#E\left(F_{q^k}\right)=q^k+1-\alpha^k-\beta^k$，其中 α,β 满足：$1-tT+qT^2=(1-\alpha T)(1-\beta T)$。

应用这个定理，可以计算许多椭圆曲线群的阶。

【例 8-4】 考虑 $F_2=Z_2=\{0,1\}$ 上椭圆曲线 E: $y^2+xy=x^3+x+1$，计算 $E\left(F_{2^n}\right)$。

解 容易验证点（0,1），O 在曲线上，所以 $E(F_2)=2$。于是 $t=q+1-\#E\left(F_q\right)=2+1-2=1$。

又 $1-tT+qT^2=2T^2-T+1=\left(1-\frac{\sqrt{7}}{2}T\right)\left(1-\frac{1+\sqrt{7}}{2}T\right)$，

所以 $\alpha=\frac{1-\sqrt{7}i}{2},\beta=\frac{1+\sqrt{7}i}{2}$。于是：

$$E\left(F_{2^n}\right)=2^n+1-\left(\frac{1-\sqrt{7}i}{2}\right)^n-\left(\frac{1+\sqrt{7}i}{2}\right)^n。$$

如：$E\left(F_{2^4}\right)=16$，$E\left(F_{2^5}\right)=22$，$E\left(F_{2^{20}}\right)=1047376$。

定义 8-8 设 G 是一个（加法）群，$P\in G$，则如果存在正整数 k，使得 $kP=0$，则称点 P 有有限阶，$k_P=\min\{k\mid kP=0,k\in N\}$ 称为点 P 的阶。否则称 P 是无限阶的，记 $k_P=\infty$。如果 G 的每一个点都有有限阶，则称 G 是一个挠群。$G[n]=\{P\mid k_P=n,P\in G\}$ 是所有 n 阶点所成的集合，可以证明它是 G 的一个子群，称为 G 的 n 阶挠子群。

定理 8-8 设 $E[n]=E\left(\bar{F}_q\right)[n]$，那么：若 n,q 互素，则 $E[n]=Z_n\oplus Z_n$；如果 E 是超奇的，则 $E\left[p^e\right]=\{O\}$，否则 $E\left[p^e\right]=Z_{p^e}$。

【例 8-5】 设 $char\left(F_q\right)\neq 2,3$，求椭圆曲线 E/F_q: $y^2=x^3+ax+b$ 上的 2 阶点。

解 如果 $(x,y)=P\in E$，$2P=0$，则 $P=-P$，所以 $(x,y)=(x,-y)$，得到 $y=0$。解 $x^3+ax+b=0$，因为其判别式 $\Delta\neq 0$，所以它有三个不同的根，设为 x_1,x_2,x_3，则 $E[2]=\{O,(x_1,0),(x_2,0),(x_3,0)\}$ 即为所求。

【例 8-6】 设 $q=p^n$，$p>2$ 是一个素数，$q\equiv 2 \bmod 3$。$0\neq b\in F_q$，考虑椭圆曲线

E_1/F_q: $y^2=x^3+b$。

解 因为 $q\equiv 2 \bmod 3$，$x_3^1+b=x_2^3+b \Leftrightarrow x_1=x_2$。所以当 x 遍历 F_q 时，x^3+b 也遍历 F_q。于是，x^3+b 有 $\frac{q-1}{2}$ 个是（非零）平方剩余。即当 x^3+b 遍历这 $\frac{q-1}{2}$ 个平方剩余时，得到椭圆曲线上的点 $\left(x,\pm\sqrt{x^3+ax+b}\right)$，共有 $2\left(\frac{q-1}{2}\right)=q-1$ 个。另外，$O,\left(\sqrt[3]{-b},0\right)\in E$，所以 $\#E\left(F_q\right)=q+1$。于是 $t=0$。

所以 E 是超奇异的。

8.3.2 椭圆曲线的同构类

定义 8-9 设群 A,B 是 Abel 群 C 的两个子群，如果：

（1）对 $\forall c\in C$，存在 $a\in A, b\in B$，$c=a+b$；

（2）$A\cap B=\{0\}$。

则称 C 是 A,B 的直和，记为 $C=A\oplus B$。

定理 8-9 $E\left(F_q\right)=Z_{n_1}\oplus Z_{n_2}$，$n_2\mid n_1$，且 $n_2\mid q-1$。

显然，如果 n_1 是素数，则 $E\left(F_q\right)$ 是 Abel 群。

E.Waterhouse 在 1969 年给出了椭圆曲线同构等价类的数目，也就是下面的定理：

定理 8-10 设 F_q 是有限域，则定义在 F_q 上同构等价类的数目为：

$$N\left(F_q\right)=2q+2+\left(\frac{-4}{q}\right)+2\left(\frac{-3}{q}\right)$$，其中 $\left(-\right)$ 是 Legendre 符号。

特别地有：

定理 8-11 如果 $char\left(F_q\right)>3$，则：

$$N\left(F_q\right)=\begin{cases}2q+6, & q\equiv 1 \bmod 12\\ 2q+2, & q\equiv 5 \bmod 12\\ 2q+4, & q\equiv 7 \bmod 12\\ 2q\ , & q\equiv 11 \bmod 12\end{cases}$$

注意：各椭圆曲线同构，则其群也同构，但反之不成立，即椭圆曲线点群同构，未必有椭圆曲线也同构。例如：

【例 8-7】 考虑有限域 F_5 上椭圆曲线 E_1: $y^2=x^3+1$，E_2: $y^2=x^3+2$。

解 容易验证：

$$E_1\left(F_5\right)=\{O,(0,1),(0,4),(2,2),(2,3),(4,0)\}，\#E_1\left(F_5\right)=6。$$

$$E_2\left(F_5\right)=\{O,(2,0),(3,2),(3,3),(4,1),(4,4)\}，\#E_5\left(F_5\right)=6。$$

于是 $E_1\left(F_5\right)\cong E_2\left(F_5\right)\cong Z_6\left(\cong Z_2\oplus Z_3\right)$。如果 $E_1\cong E_2$，则根据同构（8-5）：存在 $u\in F_5$，使得：$u^6\overline{a}_6=a_6$（由（8-2-5）前 4 式可得 $r=s=t=0$），即：$u^6=2$。但在 F_5 中，2 是非平方剩余。所以 $u^6=2$ 是不成立的。即 $E_1\not\cong E_2$。

定义 8-10 Tr: $F_{2^m} \to F_2$，$Tr(\gamma)=\gamma+\gamma^{2^1}+\gamma^{2^2}+\cdots+\gamma^{2^{m-1}}$，称为元素$\gamma$的迹；

$$Te:\ F_{2^m} \to F_4,\ Te(\gamma)=\gamma+\gamma^{2^2}+\gamma^{2^4}+\cdots+\gamma^{2^{m-2}}。$$

定理 8-12 设$F_q= F_{2^m}$，则存在$2(q-1)$个定义在F_{2^m}上的非超奇椭圆曲线的等价类。其代表元集合为：

$$\left\{y^2+xy=x^3+ax^2+b \mid b\in F_{2^m}^*, a\in\{0,\gamma\}\right\}$$

其中$\gamma\in F_{2^m}, Tr(\gamma)=1$（如果$m$是奇数，则令$\gamma=1$即可）。

定理 8-13 如果m是奇数，则存在 3 类定义在F_{2^m}上的超奇椭圆曲线，代表类集为：

（1）$y^2+y=x^3$；

（2）$y^2+y=x^3+x$；

（3）$y^2+y=x^3+x+1$。

如果m是偶数，则存在 7 类定义在F_{2^m}上的超奇椭圆曲线，代表类集为：

（1）$y^2+\gamma y=x^3$；

（2）$y^2+\gamma y=x^3+\alpha$；

（3）$y^2+\gamma^2 y=x^3$；

（4）$y^2+\gamma^2 y=x^3+\beta$；

（5）$y^2+y=x^3+\delta x$；

（6）$y^2+y=x^3$；

（7）$y^2+y=x^3+\omega$。

其中，γ不是F_{2^m}中的三次方，$Tr(\gamma^{-4}\beta)=1$，$Tr(\gamma^{-2}\alpha)=1$，$Te(\delta)\neq 0$，$Tr(\omega)=1$。

于是根据 Weil 定理可以计算出定义在F_{2^m}上的所有椭圆曲线点群的阶。

【例 8-8】 m是奇数时，计算超奇椭圆曲线E/F_{2^m}：$y^2+y=x^3$的群阶。

解 首先考虑F_2上椭圆曲线E/F_2：$y^2+y=x^3$。易验证$E(F_2)=\{O,(0,0),(0,1)\}$，$\#E(F_2)=3$，$t=q+1-\#E(F_2)=2+1-3=0$。

令$1-tT+qT^2=1+2T^2=(1-i\sqrt{2}T)(1+i\sqrt{2}T)$，得：$\alpha=i\sqrt{2}$，$\beta=-i\sqrt{2}$。

所以，$\#E(F_{2^m})=2^m+1-(-i\sqrt{2})^m-(i\sqrt{2})^m=2^m+1=q+1$。

定理 8-14 F_{2^m}上定义的超奇椭圆曲线群和群型如下：

（1）m为奇数时：

E	m	$\#E(F_{2^m})$	群　型
$y^2+y=x^3$	奇数	$q+1$	循环群
$y^2+y=x^3+x$	$m\equiv 1,7 \bmod 8$ $m\equiv 3,5 \bmod 8$	$q+1+\sqrt{2q}$ $q+1-\sqrt{2q}$	循环群
$y^2+y=x^3+1$	$m\equiv 1,7 \bmod 8$ $m\equiv 3,5 \bmod 8$	$q+1+\sqrt{2q}$ $q+1-\sqrt{2q}$	循环群

（2）m 为偶数时：

E	m	$\#E(F_{2^m})$	群　型
$y^2+\gamma y=x^3$	$m\equiv 0 \bmod 4$ $m\equiv 2 \bmod 4$	$q+1+\sqrt{q}$ $q+1-\sqrt{q}$	循环群
$y^2+\gamma y=x^3+\alpha$	$m\equiv 0 \bmod 4$ $m\equiv 2 \bmod 4$	$q+1-\sqrt{q}$ $q+1+\sqrt{q}$	循环群
$y^2+\gamma^2 y=x^3$	$m\equiv 0 \bmod 4$ $m\equiv 2 \bmod 4$	$q+1+\sqrt{q}$ $q+1-\sqrt{q}$	循环群
$y^2+\gamma^2 y=x^3+\beta$	$m\equiv 0 \bmod 4$ $m\equiv 2 \bmod 4$	$q+1-\sqrt{q}$ $q+1+\sqrt{q}$	循环群
$y^2+y=x^3+\delta x$	偶数	$q+1$	循环群
$y^2+y=x^3$	$m\equiv 0 \bmod 4$ $m\equiv 2 \bmod 4$	$q+1-\sqrt{q}$ $q+1+\sqrt{q}$	$Z_{\sqrt{q}-1}\oplus Z_{\sqrt{q}-1}$ $Z_{\sqrt{q}+1}\oplus Z_{\sqrt{q}+1}$
$y^2+y=x^3+\omega$	$m\equiv 0 \bmod 4$ $m\equiv 2 \bmod 4$	$q+1+\sqrt{q}$ $q+1-\sqrt{q}$	$Z_{\sqrt{q}+1}\oplus Z_{\sqrt{q}+1}$ $Z_{\sqrt{q}-1}\oplus Z_{\sqrt{q}-1}$

于是，对于给定的 F_{2^m} 上的超奇椭圆曲线，首先应用定理 8-7，判断曲线所属的等价类，其次再根据定理 8-14 计算群的阶。

8.4 有限域 F_{2^m} 上的算术运算

8.4.1 F_{2^m} 的基及其元素的的运算

F_{2^m} 可视为 F_2 上的 m 维向量空间，于是，存在 F_{2^m} 上的 m 个元素，$\alpha_0,\alpha_1,\cdots,\alpha_{m-1}$ 所成的基，对 $\forall\alpha\in F_{2^m}$，可惟一地表示为：

$$\alpha=\sum_{i=0}^{m-1}a_i\alpha_i\text{，}\quad a_i\in\{0,1\}\text{。}$$

这时，F_{2^m} 中元素可记为 0-1 向量形式：$\alpha=(a_0,a_1,\cdots,a_{m-1})$。

特别地，存在元素 $\beta\in F_{2^m}$，使得：

$$\left\{\beta,\beta^2,\beta^{2^2},\beta^{2^3},\cdots,\beta^{2^{m-1}}\right\}$$

为 F_{2^m} 在 F_2 上的一个基，称为 F_{2^m} 在 F_2 上的正规基。

第 2 章也给出了 F_{2^m} 中元素在正规基表示下运算的性质，现在复习一下，并给出某些改进。设 $\alpha=\sum_{i=0}^{m-1}a_i\beta^{2^i}$，那么，

（1）平方运算：

$$\alpha^2=\left(\sum_{i=0}^{m-1}a_i\beta^{2^i}\right)^2=\sum_{i=0}^{m-1}\left(a_i\beta^{2^i}\right)^2=\sum_{i=0}^{m-1}a_i\beta^{2^{i+1}}=a_{m-1}+\sum_{i=0}^{m-2}a_i\beta^{2^{i+1}}$$

，即 $\alpha^2=(a_{m-1},a_0,\cdots,$

a_{m-2})，是一个循环右移。

（2）求逆运算：

由于$\alpha^{2^m-1}=1$，所以，$\alpha^{-1}=\alpha^{2^m-2}=\left(\alpha^{2^{m-1}-1}\right)^2$。

如果 m 是奇数，则 $m-1$是偶数，有$2^{m-1}-1=\left(2^{\frac{m-1}{2}}-1\right)\left(2^{\frac{m-1}{2}}+1\right)$，所以，$\alpha^{2^{m-1}-1}=\left(\alpha^{2^{(m-1)/2}-1}\right)^{2^{(m-1)/2}+1}=\left(\alpha^{2^{(m-1)/2}-1}\right)^{2^{(m-1)/2}-1+2}=\left(\alpha^{2^{(m-1)/2}-1}\right)^{2^{(m-1)/2}-1}\left(\alpha^{2^{(m-1)/2}-1}\right)^2$，可见，只须计算$\gamma^{2^{(m-1)/2}-1}$及一次乘法即可。

如果 m 是偶数，则$\alpha^{2^{m-1}-1}=\alpha^{2^{m-1}-2+1}=\alpha^{2\left(2^{m-2}-1\right)+1}=\alpha\left(\alpha^2\right)^{2^{m-2}-1}$，只须计算$\gamma^{2^{(m-2)/2}-1}$和两次乘法。

例如：在$F_{2^{155}}$中，

$2^{155}-2=2\left(2^{154}-1\right)=2\left(2^{77}-1\right)\left(2^{77}+1\right)$

$2^{77}-1=2\left(2^{38}-1\right)\left(2^{38}+1\right)+1=2\left(2^{19}-1\right)\left(2^{19}+1\right)\left(2^{38}+1\right)+1$

$2^{19}-1=2\left(2^9-1\right)\left(2^9+1\right)+1$

$2^9-1=2\left(2^4-1\right)\left(2^4+1\right)+1=2(2+1)\left(2^2+1\right)\left(2^4+1\right)+1$

可见，在$F_{2^{155}}$中，求逆需要 10 次乘法运算和若干循环移位，但循环移位运算时间可以忽略。

在第二章，计算公式是：

$2^{155}-2=2^1+2^2+\cdots+2^{154}$，所以$a^{-1}=a^{2^{155}-2}=a^{2^0}a^{2^2}a^{2^3}\cdots a^{2^{154}}$，共需要 154 次乘法，显然这不是一个数量级上的比较。

一般地，$a^{-1}=a^{2^m-2}$需要$I(m)=\left[\log_2(m-1)\right]+\omega(m-1)-1$次乘法，其中$\omega(m-1)$表示$m-1$的二进制表示中 1 的个数。

例如：$154=2^7+2^4+2^3+2$

$$I(155)=[\log_2 154]+\omega(154)-1=7+\omega\left((10011010)_2\right)-1=10。$$

（3）乘法运算：

设$a=\sum_{i=0}^{m-1}a_i\beta^{2^i}$，$b=\sum_{i=0}^{m-1}b_i\beta^{2^i}$，$c=\sum_{i=0}^{m-1}c_i\beta^{2^i}$，且$c=ab$，则：

$c=\sum_{i=0}^{m-1}\sum_{j=0}^{m-1}a_ib_j\beta^{2^i}\beta^{2^j}=\sum_{i=0}^{m-1}\sum_{j=0}^{m-1}a_ib_j\beta^{2^i+2^j}$，令：

$$\beta^{2^i}\beta^{2^j}=\sum_{k=0}^{m-1}d_{ij}^{(k)}\beta^{2^k}，\quad d_{ij}^{(k)}\in\{0,1\} \tag{8-12}$$

那么：$c_k=\sum_{j=0}^{m-1}\sum_{i=0}^{m-1}a_ib_jd_{ij}^{(k)}$。在式（8-12）两边开$2^l$次方得：

$$\beta^{2^{i-l}}\beta^{2^{j-l}}=\sum_{k=0}^{m-1}d_{i,j}^{(k)}\beta^{2^{k-l}},\tag{8-13}$$

而
$$\beta^{2^{i-l}}\beta^{2^{j-l}}=\sum_{k=0}^{m-1}d_{i-l,j-l}^{(k)}\beta^{2^k},\tag{8-14}$$

在式（8-13）中，令$k=l$，在式（8-14）中，令 k=0 比较式（8-13）、式（8-14）中β项得：对任意$0\leqslant i,j,l\leqslant m-1$，$d_{ij}^{(l)}=d_{i-l,j-l}^{(0)}$。

于是有：$c_k=\sum_{i=0}^{m-1}\sum_{j=0}^{m-1}a_ib_jd_{i-k,j-k}^{(0)}=\sum_{i=0}^{m-1}\sum_{j=0}^{m-1}a_{i+k}b_{j+k}d_{i,j}^{(0)}$。

而$\{a_{i+k}|i=0,1,2,\ldots,m-1\}$,$\{b_{j+k}|j=0,1,2,\ldots,m-1\}$分别是$a^{2^k}$，$b^{2^k}$的分量。所以，$c_0,c_1,\cdots,c_{m-1}$的计算可以使用同一个算法。

8.4.2 有限域 F_{2^n} 上椭圆曲线的倍点公式

F_{2^n}上非奇异椭圆曲线E为：$y^2+xy=x^3+ax^2+b$，$a,b\in GF(2^n)$，$b\neq 0$，表示的点集和无穷远点O。

（1）“切——割法”一般倍点公式

令：$P=(x_1,y_1)\in E$，两倍点$2P=(x_2,y_2)$公式为：

$$\begin{cases}x_2=x_1^2+\dfrac{b}{x_1^2}\\ y_2=x_1^2+\left(x_1+\dfrac{y_1}{x_1}\right)\cdot x_2+x_2\end{cases}$$

如果忽略平方运算，那么，2倍点的x-坐标仅与P的x-坐标及系数b有关，需要一次域逆运算x_1^{-1}，两次一般域乘法$x_1^{-1}\cdot y_1$，$(x_1+x_1^{-1}y_1)\cdot x_2$，一次与$b$的乘法$(x_1^{-1})^2\cdot b$。

（2）Schroppel 倍点公式

Schroppel，J.Solinas 改进了倍点公式，减少了与b的乘法：

$\begin{cases}x_2=M^2+M+a\\ y_2=x_1^2+Mx_2+x_2\end{cases}$，其中，$M=x_1+\dfrac{y_1}{x_1}$。

下面验证以上公式的正确性：

$$\begin{aligned}x^2&=M^2+M+a\\ &=\left(x_1+\frac{y_1}{x_1}\right)^2+x_1+\frac{y_1}{x_1}+a\\ &=x_1^2+\frac{y_1^2}{x_1^2}+x_1+\frac{y_1}{x_1}+a\end{aligned}$$

$$=x_1^2+\frac{x_1^3+ax_1^2+b-x_1y_1}{x_1^2}+a$$

$$=x_1^2+x_1+a+\frac{b}{x_1^2}-\frac{y_1}{x_1}+x_1+\frac{y_1}{x_1}+a$$

$$=x_1^2+\frac{b}{x_1^2}。$$

$$y_2=x_1^2+Mx_2+x_2$$

$$=x_1^2+\left(x_1+\frac{y_1}{x_1}\right)x_2+x_2。$$

以上运算需要一次域逆 x_1^{-1}，两次乘法 $M\cdot x_2$，$x_1^{-1}\cdot y_1$，但是多了一次域加法。

（3）J.Lopez-R.Dahabo 倍点公式

Julio Lopez 和 Ricardo Dahabo 对倍点公式的 y-坐标公式进行变换：

由于 $x_2=x_1^2+\frac{b}{x_1^2}$，所以，

$$y_2=x_1^2+\left(x_1+\frac{y_1}{x_1}\right)\cdot x_2+x_2$$

$$=\frac{b}{x_1^2}+\left(x_1+\frac{y_1}{x_1}\right)\cdot x_2$$

$$=\frac{b}{x_1^2}+\frac{x_1^3+x_1y_1}{x_1^2}x_2$$

$$=\frac{b}{x_1^2}+\frac{y_1^2+b+ax_1^2}{x_1^2}x_2$$

$$=\frac{b}{x_1^2}+ax_2+\frac{y_1^2+b}{x_1^2}\cdot\frac{x_1^4+b}{x_1^2}$$

$$=\frac{b}{x_1^2}+ax_2+\left(y_1^2+b\right)\left(1+\frac{b}{x_1^4}\right)。$$

于是得到新的倍点公式：

$$\begin{cases}x_2=x_1^2+\dfrac{b}{x_1^2}\\[2ex] y_2=\dfrac{b}{x_1^2}+ax_2+\left(y_1^2+b\right)\left(1+\dfrac{b}{x_1^4}\right)\end{cases}$$

而 $\frac{b}{x_1^2}=b\times\left(x_1^2\right)^{-1}$，$\frac{b}{x_1^4}=b\times\left(\left(x_1^2\right)^{-1}\right)^2$，所以，该公式需要一次域逆，一次一般域乘法，两次系数 b 的乘法，一次系数 a 的乘法。可以选择 a 为 0 或 1，避免系数 a 的乘法。可见，

如果 b 的比特是稀疏的，那么，该算法是一种较好的算法。

（4）J.Lopez-R.Dahabo 重复倍点公式

定理 8-15 设 $P=(x,y)$ 是椭圆曲线 E 上的点， $2^k P=(x_k,y_k)$， $k=1,2,\cdots,i$， $i\geqslant 2$，则： $x_k=\frac{v_k}{u_k}$， $y_k=\frac{w_k}{u_k^2}$，其中：

$$v_{k+1}=v_k^4+bu_k^4，\quad v_0=x，$$

$$u_{k+1}=(u_k v_k)^2，\quad u_0=1，$$

$$w_{k+1}=bu_k^4 u_{k+1}+v_{k+1}\left(au_{k+1}+w_k^2+bu_k^4\right)，\quad w_0=y，\quad 0\leqslant k<i。$$

证明 对 k 作归纳法。

$k=0$ 时， $x_0=x=\frac{x}{1}=\frac{v_0}{u_0}$， $y_0=y=\frac{y}{1}=\frac{w_0}{u_0^2}$，所以，定理结论成立；

设 $k=n$ 时定理结论成立，则 $k=n+1$ 时，

$$x_{n+1}=\frac{b}{x_n^2}+x_n^2=\frac{bu_n^2}{v_n^2}+\frac{v_n^2}{u_n^2}=\frac{bu_n^4+v_n^4}{u_n^2 v_n^2}=\frac{v_{n+1}}{u_{n+1}}$$

$$\begin{aligned}
y_{n+1}&=\frac{b}{x_n^2}+ax_{n+1}+\left(y_n^2+b\right)\left(1+\frac{b}{x_n^4}\right)\\
&=\frac{bu_n^2}{v_n^2}+a\frac{v_{n+1}}{u_{n+1}}+\left(\frac{w_n^2}{u_n^4}+b\right)\left(1+\frac{bu_n^4}{v_n^4}\right)\\
&=\frac{bu_n^4}{u_{n+1}}+a\frac{v_{n+1}}{u_{n+1}}+\frac{\left(w_n^2+bu_n^4\right)v_{n+1}}{u_{n+1}^2}\\
&=\frac{w_{n+1}}{u_{n+1}^2}。
\end{aligned}$$

证明完毕。

重复倍点公式算法：

输入： $P=(x,y)\in E$， $i\geqslant 2$，

输出： $Q=2^i P$。

步骤如下：

令 $V\leftarrow x^2,U\leftarrow V,W\leftarrow y,T\leftarrow b.$

对 $k=1$ 到 $i-1$ 做

$$V\leftarrow V^2+T,$$

$$W\leftarrow U\cdot T+V\cdot\left(a\cdot U+W^2+T\right)，$$

如果： $k\neq i-1$ 则 $V\leftarrow V^2,U\leftarrow U^2,T\leftarrow b\cdot U^2,U\leftarrow U\cdot V$.

令

$$U\leftarrow U\cdot V，$$

$$M\leftarrow U^{-1}\cdot\left(V^2+W\right),$$

$$x \leftarrow U^{-1} \cdot V^2,$$

$$x_i \leftarrow M^2 + M + a, \quad y_i \leftarrow x^2 + M \cdot x_i + x_i,$$

返回$Q = (x_i, y_i)$。

推论：设 P 是椭圆曲线 E 上阶大于 2^i 的点，则应用定理 8-15 的算法计算 2^iP 需要$3i-1$次一般域乘法，$i-1$次用固定常数 b 的乘法，$5i-4$次域平方。

（5）射影坐标下的加法及倍点公式

在椭圆曲线的仿射坐标方程 $y^2 + xy = x^3 + ax^2 + b$ 中令：$x = \frac{X}{Z}$，$y = \frac{Y}{Z^2}$ 得到射影坐标方程：

$Y^2 + XYZ = X^3Z + aX^2Z^2 + bZ^4$。

在射影坐标下，椭圆曲线的群运算为：

O 元为：O（1,0,0）；

负元：$-P(X,Y,Z) = (X, XZ+Y, Z)$；

加法：$(X_0,Y_0,Z_0) + (X_1,Y_1,Z_1) = (X_2,Y_2,Z_2)$，

$A_0 = Y_1Z_0^2$，$A_1 = Y_0Z_1^2$，$B_0 = X_1Z_0$，$B_1 = X_0Z_1$，$C = A_0 + A_1$，$D = B_0 + B_1$，$E = Z_0Z_1$，$F = DE$，$G = D^2\left(F + aE^2\right)$，$H = CF$，$I = C^2 + H + G$，$J = D^2B_0E + I$，$K = D^2A_0 + I$。

$X_2 = I$，

$Y_2 = HJ + F^2K$，

$Z_2 = F^2$。

同样，忽略域加法和平方运算，则需要 13 次一般域乘法，一次与常数 a 的乘法。

倍点公式：$2(X_1,Y_1,Z_1) = (X_2,Y_2,Z_2)$，其中，

$Z_2 = Z_1^2X_1^2$，$X_2 = X_1^4 + bZ_1^4$，$Y_2 = bZ_1^4Z_2 + X_2\left(aZ_2 + Y_1^2 + bZ_1^4\right)$。

忽略域加法和平方运算，则需要 3 次一般域乘法，3 次与常数 b 的乘法，一次与常数 a 的乘法。

$Z_1 = 1$时，加法公式简化为：

$$(X_0,Y_0,Z_0) + (X_1,Y_1,1) = (X_2,Y_2,Z_2)$$

$$A = Y_1Z_0^2 + Y_0, \quad E = AC, \quad X_2 = A^2 + D + E,$$

$$B = X_1Z_0 + X_0, \quad C = Z_0B, \quad Z_2 = C^2, \quad D = B^2\left(C + aZ_0^2\right),$$

$$F = X_2 + X_1Z_2, \quad G = X_2 + Y_1Z_2, \quad Y_2 = EF + Z_2G。$$

忽略域加法和平方运算，则需要 9 次一般域乘法，一次域常数 a 的乘法。

8.5 椭圆曲线上的密码体制

公钥密码体制搬到椭圆曲线上的一般步骤是：

（1）选取合适的有限域 F_q，有限域上的椭圆曲线 E，建立一个消息空间 P 到椭圆曲线群的可逆嵌入映射，将信息进行编码：

$$\pi: P\to E,\ \pi: m\mapsto P_m;$$

（2）如果 E_k，D_k 是以离散对数为基础的加、解密算法，将此算法换成 ECDLP，得到 EC 版本的加、解密算法：

$$E_k\text{: } E\to E,\ E_k\text{: } P_m\mapsto E_k(P_m),$$
$$D_k\text{: } E\to E,\ D_k\text{: } D_k(P_m)\mapsto P_m;$$

（3）反编码：

$$\pi^{-1}: E\to P,\ \pi^{-1}: P_m\mapsto m。$$

理论上，ECC 与经典的密码体制只有明、密文空间（基域）有差别。但是，对于 ECC 而言，明文的编码（embedding，嵌入）却成了一个瓶颈，或者说，现在还没有一种一般的、有效的编码方法，将明文嵌入到椭圆曲线上。

N.Koblitz 应用二次扩域性质，提出了一种明文编码方案，但此方案的局限性是明显的。

N.Koblitz 还提出了一种比较一般的编码方法——概率嵌入法：

设 $p>3$，$q=p^n$，令 $a,b\in F_q$，$4a^3+27b^2\neq 0$，k 是一个大整数，椭圆曲线为：$E_q(a,b)$ ：$y^3=x^3+ax+b$，定义：

$$M_k=\begin{cases} q/k-1, & k\mid q \\ [q/k], & k\nmid q \end{cases}，则易见：\{0,1,\cdots,M_k k\}=\{0,1,\cdots,q-1\}=Z_q。$$

设消息为 $m\in\{0,1,\cdots,M_k-1\}$，对任意元 $x\in Z_q$，都可以表示成 p 进制形式：$x=\sum_{i=0}^{n-1}a_i p^i$，记 $R(x)=(a_{n-1},\ldots,a_0)\in F_p^n=\overbrace{F_p\times\cdots\times F_p}^{n}$。由于 F_q 中任意元都可以表示成 F_p 上次数不大于 n 的多项式，所以，$R(x)$ 对应于 F_q 中惟一元素 $f(x)$，即有：$Z_q\to F_p^{\overline{m}}\to F_q$, $x\mapsto R(x)\mapsto f(x)$。编码的概率算法如下：

1）令 $i=1$；

2）$x(m,i)=f(mk+i)$；

3）检验 $x(m,i)^3+ax(m,i)+b$ 是不是模 q 的二次剩余：

  a）如果是，那么定义 y 是 x^3+ax+b 的平方根之一，令 $\pi(m)=(x,y)\in E_q(a,b)$，结束；

  b）否则，如果 $i<k$，那么令 $i=i+1$，到第 2）步。

这样，我们得到了一种明文的编码方法：$\pi: Z_q\to E_q(a,b)$。

二次剩余的检验是有限域中的一般性工作。如果有 m，对任意的 $i=1,2,\ldots,k$，$x(m,i)^3+ax(m,i)+b$ 都不是二次剩余，那么算法不能完成消息 m 的编码。对 $x\in Z_q$，这种情况发生的概率为 $\frac{\#E_q(a,b)}{2q}$，大约为 $\frac{1}{2}$，假设 EC 点的 x 坐标随机地分布在 F_q 上，那么，不能将 x 编码的概率为 $\left(\frac{1}{2}\right)^k$，当 k 充分大时，这个概率很小。

EC 编码的局限虽然给 ECC 的实现带来了相当的不便，但是，致力于椭圆曲线密码研究的学者们提出了许多既可保持椭圆曲线的优点，又回避编码了问题的密码方案。

8.5.1 Menezes–Vanstone 密码体制

1993 年，Menezes 和 S.Vanstone 提出了一种不需要 EC 编码的密码方案。

设消息为：$(m_1,m_2)\in F_q\setminus\{0\}\times F_q\setminus\{0\}$，$P\in E(F_q)$ 是椭圆曲线上的一个 n 阶点，关于 P 的 DLP 问题是困难的。

用户 A、B 选取各自的秘钥 l_A,l_B，$l_A,l_B\in\{2,3,\dots,n-2\}$；

用户 A 计算 $Q_A=l_AP$，将 Q_A 公开，用户 B 计算 $Q_B=l_BP$，将 Q_B 公开；则用户 A、B 的密钥对分别是：(l_A,Q_A)，(l_B,Q_B)。

加密过程：

（1）A 获取 B 的公钥 Q_B，计算 $(x_1,y_1)=l_AQ_B$；

（2）A 计算 $x_2=m_1x_1 \bmod q$，$y_2=m_2y_1 \bmod q$，且向 B 发送消息 $(Q_A,(x_2,y_2))$。

解密过程：

（1）计算 $(x_1,y_1)=l_BQ_A$；

（2）计算 $m_1=x_2x_1^{-1} \bmod q$，$m_2=y_2y_1^{-1} \bmod q$，得到消息 $M=(m_1,m_2)$。

这一方案对任意 $(m_1,m_2)\in F_q\setminus\{0\}\times F_q\setminus\{0\}$ 都可以编码。

注意：

（1）如果 $x_1y_1=0$，则 A 重新选取公钥 l_A，但这种情形发生的概率很小。

（2）这一方法的膨胀率是 2。

为了弥补这两个缺陷，算法可改进为：用户 B 向 A 发送信息（$(Q_A)_x,1_{(y_1/x_1)},x_2,y_2$），而不是发送消息 $(Q_A,(x_2,y_2))$，其中 $(Q_A)_x$ 表示 Q_A 的 x 分量，$1_{(y_1/x_1)}$ 表示 $\frac{y_1}{x_1}$ 的某个比特。这时，密文的膨胀率降为 1.5。

8.5.2 F_{2^m} 上非超奇椭圆曲线 Menezes–Vanstone 密码算法

目前常用的都是 F_{2^m} 上非超奇椭圆曲线：

E：$y^2+xy=x^3+ax^2+b$，$b\in F_{2^m}^*$，$a\in\{0,r\}$，r 的迹为 1。

通常取 $a=0$，这样可以在前面介绍的倍点公式中简化计算。

1．对于有限域 F_{2^m} 上的非超奇椭圆曲线的 Menezes-Vanstone 密码算法

（1）用户 A 的加密过程：A 的公钥为 Q_B，对于明文 (m_1,m_2)，

1）获取 B 的公钥 Q_B，计算 $(x_1,y_1)=l_AQ_B$；

2）向 B 发送消息 $\left((Q_A)_x, 1_{(y_1/x_1)}, m_1x_1, m_2y_1\right)$；

（2）用户 B 解密过程：对于密文消息 $\left((Q_A)_x, M_1, M_2\right)$，

1）通过 $x=(Q_A)_x$ 计算 $y=(Q_A)_y$，求得 $Q_A=(x,y)$；

2）计算出 $l_BQ_A=(x_1,y_1)$；

3）求 $m_1=M_1x_1^{-1}$，$m_2=M_2y_1^{-1}$ 得到明文 (m_1,m_2)；

（3）求取 $y=(Q_A)_y$ 的过程如下：

选取域的正规基。

1）如果 $x=0$，则 $y^2=b$。记 $y=(y_0,y_1,\cdots,y_{m-1})$，$b=(b_0,b_1,\cdots,b_{m-1})$，则 $(y_{m-1},y_0,y_1,\cdots,y_{m-2})=(b_0,b_1,\cdots,b_{m-1})$，得到惟一解：$y_0=b_1,y_1=b_2,\cdots,y_{m-2}=b_{m-1},y_{m-1}=b_0$；

2）如果 $x\neq 0$，则对方程 $y^2+xy=x^3+ax^2+b$，作变量代换 $(x,y)\to(x,xz)$，得：$z^2+z=x+a+bx^{-2}$。令 $t=x+a+bx^{-2}$，解方程 $z^2+z=t$。

令 $z=(z_0,z_1,\cdots,z_{m-1})$，$t=(t_0,t_1,\cdots,t_{m-1})$，则：

$(z_{m-1}+z_0,z_0+z_1,\cdots,z_{m-2}+z_{m-1})=(t_0,t_1,\cdots,t_{m-1})$，即得：

$$\begin{cases} z_{m-1}+z_0=t_0 \\ z_0+z_1=t_1 \\ \quad\vdots \\ z_{m-2}+z_{m-1}=t_{m-1} \end{cases}$$

易解得：

$$\begin{cases} z_0+z_{m-1}=t_{m-1}+t_{m-2}+\cdots+t_2+t_1 \\ z_1+z_{m-1}=t_{m-1}+t_{m-2}+\cdots+t_2 \\ \quad\vdots \\ z_{m-3}+z_{m-1}=t_{m-1}+t_{m-2} \\ z+=z_{m-1}=t_{m-1} \end{cases}$$

这 $m-1$ 个独立的方程再加上 $z=y_1/x_1$ 的 1 个比特就可以确定其解。

进一步可计算出 $y=zx$。

2．$F_{2^{155}}$ 上非超奇椭圆曲线 Menezes-Vanstone 密码算法的速度

由于 $F_{2^{155}}$ 有所谓优化正规基，所以，其乘法更快，需要 155 个时钟周期。而一个逆运算需要 10 次乘法，1550 个时钟周期。

采用射影坐标，椭圆曲线加法需要 13 次乘法，将 k 用二进制表示，采用“平方——乘法”方法，kP, kaP 每一个最多可以用 *Wight*(k)−1 次 *EC* 加法，155 次 EC 倍乘，1 次域逆运算及 2 次域乘法，如果令 *Wight*(k)=30，则加密一次共需要 $2\times[29\times13+155\times7+10+2]+2=2950$ 次域乘法。

如果计算机的时钟速率为 1000MHz，则 Menezes-Vanstone 加密算法的速度为：

$$\frac{310\times1000000000}{2950\times155\times1000}\approx 678\ \text{Kbits/s}。$$

8.5.3 超奇椭圆曲线的密码体制

设 m 是奇数，考虑 F_{2^m} 超奇椭圆曲线，由前面讨论可知超奇椭圆曲线有 3 个等价类：

$$E_1: \; y^2 + y = x^2$$

$$E_2: \; y^2 + y = x^3 + x$$

$$E_3: \; y^2 + y = x^3 + x + 1$$

但是，研究表明，在这些椭圆曲线上的离散对数等价于扩域 $F_{2^{4m}}$ 上的离散对数，从而存在亚指数算法。而 F_{2^n} 上离散对数的亚指数算法时间为：$e^{1.35\sqrt[3]{n}\sqrt[3]{(\ln n)^2}}$。

由于这个原因，这些椭圆曲线上的密码体制不再比 F_{2^n} 上一般密码体制更具优势。

8.5.4 EDSA ——椭圆曲线数字签名标准 ANSI X9.62

在 ANSI X9.62 中，给出了签名算法的椭圆曲线版本 ECDSA。

首先，ECDSA 描述了其主要参数。

（1）域参数：有限域 F_q，其中 $q = p$，或 $q = 2^m$，p 是奇素数；

（2）设有限域的点数为 n，为了避免 Pollard rho，和 Pohlig-Hellman 攻击，要求 n 有大的素因子，且要求 $n > 2^{160}$，使得 $n \approx q$，$\#(E(F_q)$ 几乎是素数；

（3）F_q 有两种域表示（FR）：多项式基，正规基；

（4）$Char(F_q) = 2$ 时，椭圆曲线为：$y^2 + xy = x^3 + ax^2 + b$，$Char(F_q) > 3$ 时，椭圆曲线为：$y^2 = x^3 + ax + b$，$a,b \in F_q$ 是可选的参数；

（5）EC 上点数的计算：ECDSA 介绍了多种计算 EC 上点的方法，特别描述了 EC 上的复乘法，这是 Atkin-Morain（1991 年），Lay-Zimmer（1994 年）提出的。复乘法计算 EC 上的点是一种有效方法。

（6）在 F_{2^m} 上，曲线 $y^2 + xy = x^3 - 1$，$y^2 + xy = x^3 + x^2 - 1$ 称为 Koblitz 曲线，ANSI X9.62 推荐使用 Koblitz 曲线。

ECDSA 方案如下：

设 $D = (q, FR, a, b, G, n, h)$ 为公开参数，其中：q—域元个数，FR—域表示，a,b 是 EC 的可选系数，G——基点，$n = \#(E)$，$h = n/\#(G)$，消息为 m。

（1）签名算法：

1）选取一个随机整数 k，$1 \leqslant k \leqslant n-1$；

2）计算 $Q = kG = (x_1, y_1)$（Q 是公钥，k 是私钥）。令 $x_1 \equiv r \bmod n$，如果 $r = 0$，那么重选 k；

3）计算 $k^{-1} \bmod n$；

4）计算 $\varepsilon = SHA-1(m)$，其中 $SHA-1$ 指 $Hash$ 函数标准 $SHA-1$；

5）计算 $s = k^{-1}(\varepsilon + rd) \bmod n$，如果 $s = 0$，回到第一步重选 k；

6）发送签名消息(m,r,s)。

（2）验证算法：

1）验证r,s是否为$[1, n-1]$间的整数；

2）计算$\varepsilon = SHA-1(m)$；

3）计算$w = s^{-1} \bmod n$；

4）计算$u_1 = \varepsilon w \bmod n$，$u_2 = rw \bmod n$；

5）计算$X = u_1G + u_2Q = (x_1, y_1)$，如果$X=O$，那么否决签名；否则，计算$v = x_1 \bmod n$；

6）当且仅当$v = r$，接受签名。

（3）另外还有：

1）标准推荐了三类椭圆曲线：① $GF(p)$上的伪随机椭圆曲线；② $GF(2^m)$上的伪随机椭圆曲线；③ $GF(2^m)$ 上的 Koblitz 曲线。并具体地给出了一些椭圆曲线。

2）记基点G的阶为r，$n=rf$，那么r是一个素数，f很小，对推荐的椭圆曲线，f=1，2，4；

3）标准对不可约多项式的选取给出了具体要求，并推荐使用最小 T 复杂性正规基。

以下是一典型$GF(p)$上的伪随机椭圆曲线（十六进制表示）：

$E_{p\text{-}192}$ ： $y^2 \equiv x^3 - 3x + b \pmod p$

p = 6277101735386680763835789423207666416083908700390324961279，

r = 6277101735386680763835789423176059013767194773182842284081，

b =64210519e59c80e70fa7e9ab72243049feb8deecc146b9b1，

G_x =188da80eb03090f67cbf20eb43a18800f4ff0afd 82ff1012，

G_y =07192b95ffc8da78631011ed6b24cdd573f977a11e794811。

8.5.5 其他体制的椭圆曲线版本

还有许多其他密码体制和概念的椭圆曲线版本。如：ECDLP（椭圆曲线离散对数问题），ECDH，ECELG，ECMO（ECRSA）等。下面举几个例子来说明。

（1）ECDLP——椭圆曲线离散对数问题

对于椭圆曲线E上两点P，Q，如果已知存在x满足：$Q=xP$，则求解x的问题称为椭圆曲线上的离散对数问题。对于许多 EC 上的密码体制 ECDLP 是安全的基础。

（2）ECDH——椭圆曲线 Diffie-Hellman 密钥交换体制

选取素数$p \approx 2^{180}$，椭圆曲线$E_p(a,b)$：$y^2 = x^3 + ax + b$，及$E_p(a,b)$上基点$G = (x_1, y_1)$，G的阶n是一个大的素数。

1）A 选择私钥n_A，产生公钥$Q_A = n_AG$；

2）B 选择私钥n_B，产生公钥$Q_B = n_BG$；

3）A 计算得到共享密钥$K = n_AQ_B = n_An_BG$；

4）B 计算得到共享密钥$K = n_BQ_A = n_Bn_AG$；

易见，体制的安全性基于椭圆曲线上的离散对数的安全性：

$$Q_A = xG, Q_B = yG 。$$

【例 8-9】 考虑椭圆曲线 $E_{23}(13,22)$： $y^2=x^3+13x+22$ 。

可验证 $\#E_{23}(13,22)=22$ ， $G=(10,5)\in E_{23}(13,22)$ ，且 G 的阶 $\#<G>=11$ 。

如果 A 选取私钥 $n_A=7$, 则 $Q_A=7G=(17,21)$; B 选择私钥 $n_B=5$, 则 $Q_B=5G=(22,10)$ 。

A，B 共享的密钥为 $K=n_An_BG=35G=2G=(16,5)$ 。

（3）ECELG——椭圆曲线 ElGamal 密码体制

椭圆曲线 ElGamal 密码体制如下：

设 π ： $P\to E$ ， $m\mapsto P_m$ 是明文空间到椭圆曲线的嵌入。 $G\in E$ 为椭圆曲线上的基点。

1）用户 A 选取私钥 n_A ，计算公钥 $Q_A=n_AG$ ；

2）用户 B 为了向 A 发送信息 m，选取随机数 k，向 A 发送：

$$(kG,P_m+kQ_A)\text{；}$$

3）用户 B 解密过程为： $(P_m+kQ_A)-n_A(kG)=P_m+kn_AG-n_AkG=P_m$ 。

显然，这一体制和 Menezes-Vanstone 一样具有膨胀系数 2。

【例 8-10】 考虑椭圆曲线 $E_{23}(13,22)$： $y^2=x^3+13x+22$ ， $G=(10,5)$ 。

设用户 A 的私钥为 7，则公钥为 $Q_A=7G=(17,21)$ 。

1）设用户 B 的消息 m 编码后的信息为 $P_m=(11,1)$ ，且选取了随机数 $k=13$ ；

2）用户 B 计算： $kG=13(10,5)=(16,5)$ ， $kQ_A=13(17,21)=(20,18)$ ， $P_m+kQ_A=(11,1)+(20,18)=(18,19)$ ，并向用户 A 发送消息 $((16,5),(18,19))$ ；

3）用户 A 计算 $7(16,5)=(20,18)$ ；

4）用户 A 计算： $(18,19)-(20,18)=(18,19)+(20,-19)=(11,1)$ 。

（4）ECMO——椭圆曲线 Massey-Omura 密码体制

椭圆曲线 Massey-Omura 密码体制描述如下：

设 E 是有限域 F_q 上的椭圆曲线， $N=\#E(F_q)$ ， π ： $P\to E$ ， $m\mapsto P_m$ 是明文空间到椭圆曲线群的嵌入。

用户 A 选取一个公钥 e_A ，私钥 d_A ，满足： $e_Ad_A\equiv 1 \bmod N$ ， $1<e_A<N$ ， $1<d_A<N$ 。

对于消息 m，用户 B 加密消息为 $C_m=e_AP_m$ ，用户 A 解密过程为： $P_m=d_AC_m=e_Ad_AP_m$ 。

显然这是 RSA 的 EC 版。这一方案需要计算出 E 的阶，这并不总是方便的。

【例 8-11】 设椭圆曲线，及编码后的信息同例 8-10，考虑 ECMO。

直接验证可知 $N=\#E=22$ 。

用户选取公钥 $e_A=13$ ，可计算出私钥 $d_A=17$ ， $13*17\equiv 1 \bmod 22$ 。

用户 B 加密： $C_m=13P_m=13(11,1)=(14,21)$ 。

用户 A 解密： $P_m=17C_m=17(14,21)=(11,1)$ 。

8.6 习题

1. 求有限域 Z_7 上椭圆曲线 E：$y^2+3xy+y=2x^3+6x^2+5$ 的所有点。

2. 求有限域 Z_{11} 上椭圆曲线 E：$y^2+xy+3y=x^3+10x^2+7$ 的所有点。

3. 求有限域 Z_{71} 上椭圆曲线 E：$y^2+70xy+39y=x^3+60x+17$ 的所有点。

4. 计算有限域 F_2^{32} 上椭圆曲线 E：$y^2+xy=x^3+x+1$ 的点数。

5. 计算有限域 F_5^{64} 上椭圆曲线 E：$y^2+xy=x^3+2x+1$ 的点数。

6. 计算有限域 F_5^{64} 上椭圆曲线 E：$y^2+xy+y=x^3+4x^2+x+3$ 的点数。

7. 验证 $A=(7,10)$，$B=(15,18)$ 是有限域 Z_{19} 上椭圆曲线 E：$y^2+7xy+y=x^3+15x$ 的点，求 $A+B$，$11A$。

8. 验证 $A=(9,11)$，$B=(32,36)$ 是有限域 Z_{37} 上椭圆曲线 E：$y^2+11xy=x^3+17x^2+x+20$ 的点，求 $A+B$，$25A$。

9. 验证 $A=(18,98)$，$B=(104,167)$ 是有限域 Z_{179} 上椭圆曲线 E：$y^2+127xy+11Y=x^3+8x^2+31x+9$ 的点，求 A，B 的阶。

10. 验证 $A=(80,83)$，$B=(161,142)$ 是有限域 Z_{173} 上椭圆曲线 E：$y^2+11xy=x^3+17x^2+x+20$ 的点，求 $A+B$，$25A$。

11. 设有限域 Z_{103} 上椭圆曲线 E：$y^2+26xy+57y=x^3+3x^2+79x+100$，验证 $P=(2,6)$ 是 $E(Z_{103})$ 上的点，实现 Menezes-Vanstone 密码体制。

12. 设有限域 Z_{563} 上椭圆曲线 E：$y^2+26xy+57y=x^3+3x^2+79x+100$，验证 $P=(242,244)$ 是 $E(Z_{563})$ 上的点，实现 Menezes-Vanstone 密码体制。

13. 验证 $P=(111,59)$ 为有限域 Z_{193} 上椭圆曲线 E：$y^2+43xy+91y=x^3+68x^2+137$ 上的点。以 P 作为基点，构造 Menezes-Vanstone 密码体制：

（1）用户 A 的公钥为 $Q_A=l_aP=(2,156)$，B 的秘密密钥 $l_B=87$。用户 B 的公钥解密密文 $c=(Q_A,(114,149))$。

（2）用户 A 的公钥为 $Q_A=l_aP=(11,165)$，B 的秘密密钥 $l_B=78$。用户 B 的解密密文 $c=(Q_A,(160,129))$。

14. 求有限域 Z_{149} 上椭圆曲线 E：$y^2+77xy+32y=x^3+28x^2+91x+45$ 的点 $P=(23,58)$ 的阶。以 P 作为基点，构造 Menezes-Vanstone 密码体制，用户 A，B 分别选择秘密密钥 $l_A=100$，$l_B=75$，分别计算用户 A，B 的公钥 $Q_A=l_aP$，$Q_B=l_BP$，和 A，B 的共享密钥 l_Al_BP。加密消息 $m=(111,49)$，并进行解密验证。

15. 求有限域 Z_{149} 上椭圆曲线 E：$y^2+3xy+2y=x^3+80x^2+133$ 的点 $P=(98,14)$，的阶。

16. 选取有限域 Z_{1229} 上椭圆曲线 E：$y^2+1017xy+1011y=x^3+8x^2+831x+79$ 及其上点 $G=(330,922)$：

（1）G 作为基点。构造 ElGamal 密码体制。加密消息 $P_m=(707,1208)$，并进行解密验证

（2）设用户 A 的私钥为 n_A=999，公钥 Q_A=n_AG=（508，1124），已知密文为 P_c=（kG，P_m+kQ_A）=（(100, 499)，(242, 597)），解密密文。

（3）设用户 A 的私钥为 n_A=793，公钥 Q_A=n_AG=(717，202)，已知密文为 P_c=(kG，P_m+kQ_A) =（(1184, 1225)，(1206, 962)），解密密文。

17．构造有限域 F_2^8 及其一个正规基。

18．证明 ECDSA 的验证原理。

第 9 章　混沌理论在密码学中的应用

9.1　混沌的基本概念

混沌理论与微积分不同，它是一门专门研究奇异函数、奇异图形的数学理论，研究自然界有序、无序间规律的学科。1974 年，美国青年物理学家 Feigenbaum 发现了普适性常数；1975 年，李天岩和其导师 Yoke 发表了著名论文“Period Three Implies Chaos”，首次使用了“混沌”（chaos）这个名词。混沌理论已被广泛地应用于气象预报、生态模型、经济预测、地震预报等等领域，被认为是二十世纪与相对论、量子力学并列的三大发现之一。

混沌系统有两个主要特征：对初始条件的敏感性和系统变化的不可预测性，这两个特性恰恰是密码学随机序列的重要特征，所以，混沌理论也被加入到密码学基础理论之中。

混沌映射的奇异表现提供了一种随机序列的产生方法。

下面给出混沌理论的基本概念。

首先给出周期点的概念。

定义 9-1　设 f: $A \to A$ 是一个自映射，对 $x \in A$，如果存在正整数 n，满足：

$f^n(x) = \underbrace{f(f(\cdots f}_{n\ f}(x)\cdots)) = x$，但对任意正整数 k，$k<n$，$f^k(x) \neq x$，

则称 x 是 f 的一个（真）n 周期点，n 称为周期。

定义 9-2　设 f: $I \to I$ 是闭区间 I 上的一个连续自映射，如果它满足下列条件，则称它在区间 I 上是混沌的：

（1）f 存在任意周期的周期点；

（2）存在 I 上的不可数子集 S，满足：

1）对 $\forall x, y \in S$，$x \neq y$，有：$\overline{\lim_{n\to\infty}}\left|f^n(x) - f^n(y)\right| > 0$；

2）对 $\forall x, y \in S$，$x \neq y$，有：$\underline{\lim_{n\to\infty}}\left|f^n(x) - f^n(y)\right| = 0$；

3）对 $\forall x \in S$，及 f 的任意一个周期点 y，有：$\overline{\lim_{n\to\infty}}\left|f^n(x) - f^n(y)\right| > 0$。

条件（1）说明，混沌现象中，各种周期成分并存，而条件（2）表明，非周期成分也共存；（2）中条件 2），3）表示任何两点的迭代轨迹时分时合；3）表示非周期轨道不可能呈现出渐近的周期行为。这些条件充分地反映了映射迭代轨迹的混乱（随机）特性。

李—Yoke 定理给出了一个判别映射混沌的条件：周期 3 意味着混沌。数学表示为：

定理 9-1　（李—Yoke 定理）：设 f: $I \to I$ 是闭区间 I 上的一个连续自映射，若 f 有 3 周期点，则它就有任意周期的周期点，且 f 在上述意义下是混沌的。

关于 2 维，3 维，n 维混沌映射有更加丰富的内容，例如：Smale 马蹄、横截同宿轨道、李雅普诺夫指数等。

一个最简单的混沌系统由 Logistic 函数：

$$y=\lambda x(1-x),\quad x\in[0,1]$$

的迭代序列 $x_{n+1}=\lambda x_n(1-x_n)$ 给出。

Feigenbaum 研究迭代轨道倍周分岔达到混沌的过程时发现，分岔速度是一个常数。例如，对于 Logistic 函数 $y=\lambda x(1-x)$给出的迭代映射 $x_{n+1}=\lambda x_n(1-x_n)$，当参数$\lambda$达到

$\lambda_1=3$，$\lambda_2=3.449490$，$\lambda_3=3.544090$，$\lambda_4=3.564407$，…，时，迭代序列开始出现 2，2^2，2^3，2^4，…，周期点，即：

当 $1\leqslant\lambda<\lambda_1$ 时，$\{x_n\}$有 1 个收敛点，

当$\lambda_1\leqslant\lambda<\lambda_2$ 时，$\{x_n\}$由 1 个收敛点分岔为 2 个收敛点，

当$\lambda_2\leqslant\lambda<\lambda_3$ 时，$\{x_n\}$由 2 个收敛点分岔为 2^2 个收敛点，

当$\lambda_3\leqslant\lambda<\lambda_4$ 时，$\{x_n\}$由 2^2 个收敛点分岔为 2^3 个收敛点，

当$\lambda_4\leqslant\lambda<\lambda_5$ 时，$\{x_n\}$由 2^3 个收敛点分岔为 2^4 个收敛点，

…………，

并发现λ_k几乎以几何级数变化，即有：

$$\delta_k=\frac{\lambda_{k+1}-\lambda_k}{\lambda_k-\lambda_{k-1}}\xrightarrow{k\to\infty}4.66920166091029909\cdots。$$

Feigenbaum 还发现，对于像 $y=\lambda x(1-x)$，$\lambda e^{x(1-x)}$，$\lambda\sin(\pi x)$，$x(1-\lambda(1-x))$，$x\in[0,1]$等单峰函数都有这一现象，且这个极限值是相同的。大多数学者认为，这个极限是一个普适性常数，其重要性不亚于π，e 等。人们将其称为 Feigenbaum 常数。这一常数的一个重要用途是可以帮助我们预测达到混沌的时间。

Feigenbaum 分岔现象如图 9-1 所示。

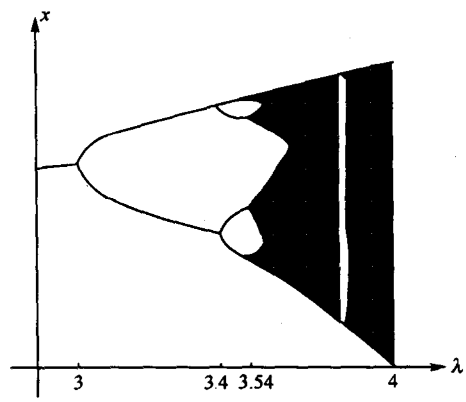


图 9-1　Feigenbaum 分岔图

9.2　混沌序列的产生及其随机序列

一次一密的密码体制中，任意长随机比特序列的生成、传输、存储是其无法逾越的瓶颈。混沌映射被用来产生（伪）随机序列，较好地地解决了这一问题，从而为一次一密密码体制的应用提供了基础。

混沌序列的应用中，最大的困难和要求是其随机性度量和事实周期的研究。可以断言，混沌序列不可能严格满足 Golomb 随机性条件，也许根本不满足 Golomb 公设。由于计算精度的限制，混沌序列实际上是周期的，对密码学应用而言，这个周期必须足够大，但是，我们甚至不能计算一个混沌系统的事实周期。这给混沌系统的密码学应用带来了巨大的障碍，也成为密码学研究的主要内容。

Golomb 随机性并不是度量随机性的惟一标准，事实上，混沌系统的混沌特性本身就是一种随机性。对于迭代混沌序列：x_1，x_2，…，x_n，…也可以用分布来刻画其随机性行为。例如，二次映射 $y=1-2x^2$ 生成的迭代混沌序列的在[−1,1]区间上的分布密度函数为：

$$\rho(x)=\frac{1}{\pi\sqrt{1-x^2}}，\quad x\in[-1,1]。$$

对于一般的混沌迭代函数很难得到精确的分布密度的解析公式。工程实践中，通常用数值分析方法来得到大致的分布密度。

1989 年，Robert Matthews 在 Logistic 函数的基础上，首次给出了用于加密的随机序列的生成函数：

$$g(x)=(\beta+1)(1+\frac{1}{\beta})^{\beta}x(1-x)^{\beta}\ ,1\leqslant\beta\leqslant 4\ 。$$

研究表明，在参数β的以上范围内，迭代映射 $x_{n+1}=g(x_n)$是混沌的。迭代值对初始值 x_0 是敏感的。

如果设参数β和初始值 x_0 取 D 位十进制数，则密钥空间的大小为 10^{2D}，转换成二进制，令：

$$2^{H(K)}=10^{2D}，得：H(K)=6.6D。$$

可见，每个密钥有 $6.6D$ 比特。如果计算机的运算速度为每秒 R 次，则对密钥穷举攻击所需时间为：$Y=\dfrac{10^{2D}}{10^{7.5}\times R}$ 年。

如果 D=10，$R=10^6$，则：$Y=3.17098\times10^6$ MIPS。

显然，可以认为是安全的。并且，我们也看到，只要简单地提高数据有效位数，攻击时间就以指数速度增长。

但是，Daniel D Wheeler 指出，由于计算的精度问题，以上映射实际是循环的，并且，循环周期在 149 至 16456 之间，显然，用这样的序列作为密钥是不安全的。于是，Douglas W.Mitchell 又提出一种非线性函数映射：

$$x_n=an^b，\ a>0，\ b>1。$$

但随后 Daniel D Wheeler 指出，对于这种映射生成的序列，具有差分的密码分析方法，因而也是不安全的。

Daniel D Wheeler，Robert A.J.Matthews 进一步研究得出，只要提高计算机算术运算的精度，Matthews 的混沌序列生成方法仍然适用于密码学。

考虑一个 2 维 Hénon 映射：$T(x_i,y_i)=(x_{i+1},y_{i+1})$，

$$\begin{cases} x_{i+1} = y_i + 1 - 1.4x_i^2 \\ y_{i+1} = 0.3x_i \end{cases}$$

对于任意给定的初始点（x_0，y_0），在这个映射的迭代下，无限地趋于一个无穷集，这个无穷集称为奇异吸引子。

设 A（-1.33，0.42），B（1.32，0.133），C（1.245，-0.14），D（-1.06，-0.5）如果选取的初始点在矩形 ABCD 中，则迭代序列的所有点都包含在该矩形中。

用 Hénon 映射来产生伪随机序列是应用了奇异吸引子的特殊性质。理论上，Hénon 映射产生的序列（简称 Hénon 序列）是永不重复的，且对初始点具有非常的敏感性，迭代轨迹随机地分布在吸引子附近（如果初始点正好在吸引子中，则迭代轨迹将都在吸引子中）。以上这些特点正是密码学所需的随机性。

首先，需要将随机序列转化为二进制比特序列。完成这一工作的方法有任意多种，但是，我们的方法应该使得序列中 0，1 的个数很接近，尽量满足 Golomb 公设。

设迭代序列是：$\{(x_i,y_i)|i=1,2,\ldots\}$，选取中值 z 满足：

$$\text{Prob}\{x_i \leqslant z\} = \text{Prob}\{x_i > z\},$$

定义二进制序列$\{s_i\}$为：

$$s_i = \begin{cases} 0, x_i \leqslant \overline{x} \\ 1, x_i > \overline{x} \end{cases}$$

对 ABCD 内的 237 个初始点，长度为 1000 的 Hénon 序列的研究发现 $z_x \approx 0.39912$。类似地，可以对纵坐标定义二进制序列，得出 $z_y \approx 0.11977$。也可以考虑线性组合：ax_i+by_i，确定其相应的中值。

对二进制 Hénon 序列的理论分析可能是很困难的，所以，一般只能用实验来检测它们对初始点的依赖性、序列相关性、比特熵、分布等性态。

理论上，矩形 ABCD 中的任意点都可以作为初始点，因而初始点的数目是无限的。但是，计算机只能处理有限数集，并且计算机的浮点数据精度是有限的，位置不同但是非常接近的初始点可能得到相同的序列，有时虽然序列不完全相同，但是，序列的前面许多点是相同的。因而，无论是否完全相同，实际工作中需要丢弃前面的某些比特。

实验显示，如果初始点的差异为$\Delta x=10^{-19}$，并丢弃前 2000 个比特后，两个序列的相关性就接近 0，可以达到密码学的要求。此时，密钥空间的大小（初始点的个数）为：

#（K）$\approx 1.4764\times10^{36} \approx 2^{126}$，达到 126 比特。

二进 Hénon 序列的自相关性检测却不理想，两个临近初始点的序列间存在重要的统计相关性。消除这种相关性的方法是将两个或多个二进 Hénon 序列按比特进行异或来产生一个具有更好统计性质的新序列。初始点在 ABCD 中严格随机选取的 k 个二进 Hénon 序列的异或所成序列称为 k 阶二进 Hénon 序列。实验结果说明这种序列的相关性接近 0，具有一致的 n—元分布，和接近 1 的比特熵。这种序列似乎是适合密码学要求的，但是，真正地应用于密码学，还需要进行进一步的理论和实验研究。

9.3 逆混沌密码体制

以上讨论的是混沌序列在序列密码学中的应用，下面介绍一个基于混沌逆映射的对称密码方法。

设函数 $y=f_\lambda(x)$，$x\in[0,1]$，$\lambda\in[a,b]$确定的迭代映射 $x_{n+1}=f(x_n,\lambda)$是混沌的，且 $y=f_\lambda(x)$有有限支反函数。

例如：Logistic 映射$f_\lambda(x)=\lambda x(1-x)$，帐篷映射 $f_\lambda(x)=\begin{cases}\dfrac{1}{\lambda}x, & 0\leqslant x\leqslant\lambda\\ \dfrac{x-1}{\lambda-1}, & \lambda<x\leqslant 1\end{cases}$ 等有两支反函数，那么，可以由此构造一种对称密码体制。

令 $K=[a,b]$为密钥空间，$P=[0,1]$为明文空间，选择适当的正整数 n。

设待加密消息为 $m\in P$，则：

加密算法为：

$$c=\underbrace{f_\lambda^{-1}(f_\lambda^{-1}(\cdots f_\lambda^{-1}}_{n次}(m)\cdots))=f_\lambda^{-n}(m),$$

其中 f_λ^{-1} 为f_λ的任意一支反函数，n 次迭代中，每次随机地选取一支。可见，如果 f_λ^{-1} 有 s 支，则就有 s^n 个正确的密文版本，这是一种概率密码体制。

解密算法为：

$$m=f_\lambda(f_\lambda(\cdots f_\lambda(c)\cdots))=f_\lambda^{n}(f_\lambda^{-n}(m)),$$

即，以密文为初始值，作 n 次迭代。易见解出的明文是惟一的。

迭代次数 n 一般要尽量地大，n 越大，差别不大的明文加密后得到的密文的差别就越大。此外，n 的选取还需要使明、密文满足一定的分布性要求，可用如下方法来选取：

（1）随机地选取一个点作为密文，取一些密钥进行解密，考察明文的分布，使之相对于密钥在[0,1]上一致地分布。

（2）将（1）中选取的密钥稍加变化，再解密密文，得到新明文的分布，这个分布与（1）中分布独立。

如果 n 的选取满足（1），（2），则体制就会具有良好的安全性。

混沌技术在保密通信的应用中已取得了良好的效果，在密码学中的应用还有待进一步研究。

9.4 示例

（1）设选取的混沌函数为：$f(x,\lambda)=\lambda x(1-x)$，则两支逆混沌函数为：$f_{\pm}^{-1}(m,\lambda)=\dfrac{1}{2}\pm\sqrt{\dfrac{1}{4}-\dfrac{m}{\lambda}}$，令 $n=10$。

（2）设明文为：$m=0.791878$；

（3）加密算法为：$E(m,\lambda)=f^{-10}(m,\ \lambda)$，可以产生 $2^{10}=1024$ 个可能的密文。

例如：$E(m, \lambda)=f^{-10}(0.791878,4)=\{$
0.0255731, 0.983993, 0.999796, 0.201291, 0.0288681, 0.158915, 0.128187, 0.0236716, 0.98741, 0.997393, 0.498003, 0.889698, 0.0972101, 0.742839, 0.000508285, 0.966881, 0.770987, 0.00770645, 0.930978, 0.443824, 0.104602, 0.130245, 0.996834, 0.184349, 0.00251379, 0.678948, 0.999948, 0.114764, 0.0012771, 0.358824, 0.593482, 0.498929, 0.995716, 0.442905, 0.193961, 0.0396601, 0.969042, 0.992294, 0.112231, 0.830405, 0.231597, 0.330547, 0.288029, 0.128806, 0.203756, 0.190813, 0.932988, 0.587447, 0.793767, 0.562268, 0.0458632, 0.335458, 0.550075, 0.566229, 0.994744, 0.879303, 0.0883124,0.998933,0.803607,0.000125446,0.875888, 0.877297, 0.0905916, 0.993964,…………}。

$E(m, \lambda)=f^{-10}(0.791878,3.9999933)=\{$
0.101434, 0.407428, 0.488819, 0.999449, 0.0147253, 0.174927, 0.984754, 0.620449, 0.9129, 0.626395, 0.344178, 0.969361, 0.55922, 0.97215, 0.0342265, 0.298361, 0.100876, 0.0542892, 0.867053, 0.0299046, 0.931446, 0.519464, 0.955029, 0.967211, 0.105169, 0.179614, 0.887183, 0.00825241, 0.0327892, 0.177265, 0.941898, 0.987615, 0.997784, 0.978158, 0.0227479, 0.116135, 0.445958, 0.270683, 0.03501, 0.689501, 0.655822, 0.25262, 0.926238, 0.379551, 0.0674753, 0.486674, 0.0705863, 0.0268491, 0.326796, 0.755337, 0.803608, 0.733935, 0.162743, 0.0900615, 0.00389254, 0.296402, 0.913421, 0.416489, 0.470416, 0.3745, 0.634387, 0.55007, 0.959979, 0.0132836, …………}。

（4）解密算法为：$D(m, \lambda)=f^{10}(m,\lambda)$，不难验证：
$D(m, \lambda)= D(f^{-10}(0.791878,4), 4)= 0.791878$；
$D(m, \lambda)= D(f^{-10}(0.791878,3.9999933), 3.9999933)= 0.791878$。

混沌算法一般都用连续函数表示，所以，要将其真正应用于加密，计算中的数值处理、精度问题等可能成为主要工作。

9.5 实验题

1．观察迭代数列 $x_0=0.4$，$\beta=3$，$x_n=(1+\beta)(1+1/\beta)x(1-x)^{\beta}$的周期，精确到小数点后 6 位。

2．Hénon 映射 T：$(x_i,y_i)=(x_{i+1},y_{i+1})$，为

$$\begin{cases} x_{i+1} = y_i + 1 - 1.4x_i^2 \\ y_{i+1} = 0.3x_i \end{cases}$$

取 $z_x = 0.39912$， $z_y = 0.11977$， $3z_x + 5z_y = 1.79621$， 按

$s_i = \begin{cases} 1, & 3x_i + 5y_i > 1.79621 \\ 0, & 3x_i + 5y_i \leqslant 1.79621 \end{cases}$， $i = 1, 2, \cdots$ 构造序列 $\{s_i\}_{i=1,2,\cdots}$，观察序列的周期，精确到小数点后 6 位。

第 10 章　量子密码理论

10.1　引言

1969 年春，Stephen Wiesner 首次提出“共轭编码”（Conjugate coding）的概念，稍后，SIGACT 新闻（SIGAT NEWS）发表了 Stephen Wiesner 的文章，并首次报道了有关量子密钥分配的实验。

1984 年，Bennett. Charles H., Brassard. Gille 基于 Wiesner 的思想，首次提出了基于量子理论的编码方案及密钥分配协议，现称为 BB84 协议。

BB84 协议的基于两个量子状态的简化协议，称为 B92 协议。

1989 年 SIGACT News 报道了第一个量子密码实验，IBM 和美国 Montreal 大学联合实验，在 32 厘米距离实现了随机比特串的安全传输。

1993 年，英国 BT（British Telecom）实验室的 Paul D.Townsed 和 Christonphe Marand，John Raarity，Paul Tapster，Ian Thompson 等人在 30 公里的光纤信道上成功地实现了 BB84 密钥交换协议。但是这还是在实验室条件下进行的，30 公里的光纤盘卷在一起，收发两端都在同一个房间。

1995 年，Townsed 和协作者们进行了一个网络通信量子密码的实际实验。Geneva 大学的 Nicolas Gisin 与 J.Breguet，Antoine Muller，Hugo Zbinden 第一次成功地实现了一次真实的量子密码通信，收发两端分别在 Nyon 和 Geneva，两地相距 23 公里，作为量子通道的光纤穿过了 Geneva 湖。

据推测，BB84 协议至少可以在 100 公里范围实现。

量子密码的安全性基于量子力学中的 Heisenberg 测不准原理。可以说，谁攻破一个基于整数因子分解的密码体制，则他应该找到了一个因子分解的有效算法。而谁要攻破一个量子密码协议，除非他能够否定量子力学定律，所以说，量子密码是一种绝对安全的密码技术。

10.2　Heisenberg 测不准原理

Heisenberg 测不准量子原理是量子密码的基本依据。采用 Dirac 记号，这个原理可表述为：

设 A，B 是量子的两个算子的测量值，$\langle A\rangle$，$\langle B\rangle$ 表示算子的理论值，则：

$$(\Delta A)^2(\Delta B)^2 \geqslant \frac{1}{4}[A,B]^2,$$

其中 $\Delta A = A-\langle A\rangle$，$\Delta B = B-\langle B\rangle$，$[A,B]=AB-BA$。

可见 $(\Delta A)^2$，$(\Delta B)^2$ 刻划了测量值的不确定性，它们相互制约，$[A,B]\neq 0$ 时，如果

$(\Delta A)^2$ 减少，则 $(\Delta B)^2$ 就增加，相反，如果 $(\Delta A)^2$ 增加，则 $(\Delta B)^2$ 就减少，这就是说，不可能同时准确测量两个量 A，B。

在 Hilbert 空间（完备的，带有内积的，线性空间）H 中，量子就是空间的向量。设 $\{\varphi_i\}$ 为 H 的一个标准正交基，则 H 中任何一个向量 φ 都可以表示成基向量 φ_i 的正交级数：

$$\varphi = \sum_{i=0}^{\infty} \lambda_i \varphi_i \text{。}$$

通常，$\{\lambda_i\}$，$\{\varphi_i\}$ 是 H 中一个对称线性变换 A 的特征值、特征向量构成，即有：$A\varphi_i=\lambda_i\varphi_i$。

而 A 就是量子测量算子，A 的特征值、特征向量称为本征值、本征态。于是以上说明用量子理论术语可描述为：

任何一个量子（态）φ，都可以展开成量子算子 A 的本征态 φ_i 的正交级数：

$$\varphi = \sum_{i=0}^{\infty} \lambda_i \varphi_i \text{。}$$

对量子态 φ 进行测量时，φ 将以概率 λ_i 被测量为 φ_i。如果 $\varphi=\varphi_i$，则 φ 以概率 $\lambda_i=1$ 测得为 φ_i，这就是量子密码的基本原理。

10.3 BB84 协议

量子密码应用中有密钥分配、比特承诺等协议，但比特承诺等协议已被证明是不可行的。下面描述 BB84 的密钥分配协议。

BB84 协议是用光量子来描述的。

光量子的振动方向是任意的，如果所有光子都沿着同一个方向振动，则称其为偏振光。偏振滤光器只允许某一方向偏振光子通过，或者说其他方向的偏振光子以一个概率转移到偏振器的方向，也就是说，如果光子的偏振方向与滤光器的方向一致，或夹角很小，则光子通过滤光器的概率很大，夹角为 90°时概率最小。BB84 协议就是通过偏振滤光器来发送和接受光量子而实现密钥分配的。

设 A，B 要通过光子脉冲来传递密钥，他们首先要约定编码方法，考虑四类偏振光子—，╱，|，╲，即偏振角为 0°，45°，90°，135°的光量子，编码如表 10-1。

表 10-1 编码方法

符　号	Bit
—，╱	0
|，╲	1

也可以采用其他的编码方法，这种编码方法是公开的。

设想，如果 A 按照编码发送一个光子，B 可以分别用四种偏振角的滤光器来测试，根据光子能够通过哪一个滤光器就可以确定光子的偏振角，从而得到 A 发送的信息。但是，测不准原理说明 B 不可能测量光子的两个量，偏振器的测量会改变光子的偏振角。如果 A 发送一个偏振态为“-”的光子（信息 0），B 设置滤光器的偏振角为 0°，那么 B 将测得一

个光子，从而得到信息 0；如果 B 设置滤光器的偏振角为 45°，则 B 只能得到一个随机值，且光子的偏振态也将改变。这时，即使 B 再改变滤光器的偏振角为 0°，也不一定能得到信息 0。这就是为什么量子密码协议能够发现窃听行为的基本原理。

下面通过一个例子来阐述密钥分配协议。

10.3.1 无窃听量子信道的密钥分配过程

协议分两个阶段：

（1）量子传输

1）首先 A 随机地生成一个比特流，将要共享的密钥流在其中产生，这也是保密的。

2）根据编码方法发送偏振光子脉冲。相应于比特 0，可以发送偏振光子—，或／，相应于比特 1，可以发送偏振光子｜，或＼。

3）B 设置其接收滤光器序列。由于 B 不知道 A 的设置，所以只能随机地设置，对于每一个位置，B 设置正确的概率为 1/4。

4）B 读取光子序列，并记录相应比特信息。

（2）公开协商

1）B 公布其滤光器序列的设置。

2）A 对照自己的设置，指出 B 设置正确的位置。

3）B 选取正确位置的比特信息。

4）为了检验信道的窃听行为，B 公布部分选定的比特，一般为 1/3。

5）A 检查 B 公布的比特的正确性，如果没有窃听行为，这些比特应该是相同的。

6）双方约定，用剩余的 2/3 比特作为共享密钥，这样就完成了一次秘密密钥共享过程。

对于一个位置 B 设置正确的概率为 1/4，所以，如果 A 发送 n 比特，则 A，B 平均可以得到 $n/2$ 比特，再减去 B 公布的 1/3 检验比特，最后 A，B 平均可以得到$(n/4)\times(2/3)= n/6$ 个共享比特。

下面是一个具体的例子：

（1）量子传输

A 生成随机比特流	0	1	1	1	0	0	0	1	0	1	1	0	0	1	1	0	1	1	1	0
A 按流发送光子脉冲	／	＼	｜	｜	／	—	—	＼	／	＼	｜	—	／	＼	｜	—	｜	｜	＼	／
B 随机设置滤光器	｜	＼	／	—	／	—	＼	／	｜	＼	／	—	／	／	｜	｜	／	｜	／	／
B 读取比特流	1	1	1	0	0	0	1	1	0	1	0	0	0	1	1	1	0	1	0	0

（2）公开协商

B 公开其偏振器的设置	｜	＼	／	—	／	—	＼	／	｜	＼	／	—	／	／	｜	｜	／	｜	／	／
A 确定正确的设置		√			√	√				√		√	√		√			√		√
B 确定正确比特		1			0	0				1		0	0		1			1		0
B 公布部分检验比特		1				0							0							
A 检验		√				√							√							
最后 A,B 共享的比特流					0					1		0			1			1		0

10.3.2 有窃听量子信道的密钥分配过程

与无窃听过程一样，有窃听密钥共享过程也有两个阶段，但是，在 A，B 之间插进了第三者 C，由于 C 的插入，改变了光量子的状态，从而，在 A，B 的讨论中可以发现 C 的存在。

下面以具体实例来说明

（1）量子传输与窃听

A 生成随机比特流	0	1	1	1	0	0	0	1	0	1	1	0	0	1	1	0	1	1	1	0
A 按流发送光子脉冲	／	＼	｜	｜	／	—	—	＼	／	＼	｜	—	／	＼	｜	—	｜	｜	＼	／
C 随机设置滤光器	｜	—	／	—	／	—	＼	／	｜	＼	｜	—	／	／	＼	｜	／	＼	／	—
C 读取比特流	1	1	1	0	0	0	1	1	0	1	1	0	0	1	1	1	0	1	0	0
B 设置滤光器	／	—	／	＼	／	—	＼	＼	｜	＼	｜	＼	／	—	｜	／	／	＼	／	／
B 读取比特流	1	1	1	0	0	0	1	0	0	1	1	1	0	1	1	0	0	1	0	0

（2）公开协商

B 公开其偏振器的设置	／	—	／	＼	／	—	＼	＼	｜	＼	｜	＼	／	—	｜	／	／	＼	／	／
A 对照发送的光子脉冲	／	＼	｜	｜	／	—	—	＼	／	＼	｜	—	／	＼	｜	—	｜	｜	＼	／
A 指出 B 正确的设置	√				√	√		√		√	√		√		√					√
B 对照所获得的比特流	1	1	1	0	0	0	1	0	0	1	1	1	0	1	1	0	0	1	0	0
B 选出正确的比特	1				0			0		1	1		0		1					0
B 公布部分比特	1				0			0												
A 检验	×				√			×												

可见，如果没有 C 的窃听，A 的检验完全正确，那么，剩下的比特即可成为 A，B 的共享密钥。反之，如果存在 C 的窃听，一般通过讨论都可被发现。

考虑有窃听时检验通过的概率：

假设 A 发送 n 比特，C 设置正确的概率都为 1/4。

B 公布的比特必须是 A，B，C 三者的设置一致时才可能发生。

对于一个位置，A，B，C 三者的设置一致的概率为 1/16。

A，B 设置一致的概率也是 1/4，所以，B 平均可以得到 $n/4$ 个正确设置。

所以，如果 B 公布 1/3 比特，则 B 平均需要公布$(n/4)\times(1/3)=n/12$ 比特进行检验。这 $n/12$ 个比特能够通过检验的概率为：

$$P=\left(\frac{1}{16}\right)^{\frac{n}{12}}=\frac{1}{2^{n/3}},$$

如果需要得到 64 比特的密钥，则至少有 $n=6\times64=384$，这时 $P=\frac{1}{2^{384/3}}=\frac{1}{2^{128}}$ $=2.938734\times10^{-20}$，可见 C 的窃听不被发现的概率几乎是 0。

以上的分析说明，量子密码的安全性是一种绝对安全性，可以抗击具有无穷计算能力的攻击者。有人认为，量子密码学可能是未来惟一安全的密码技术。但是，量子密码走向实用

还有许多难题要解决，这些将是未来量子密码技术研究的重要内容。

10.4 习题

1．如果发送 A，B，C，D 四种粒子的概率分别为 P_A，P_B，P_C，P_D，X 粒子通过 Y 转发器时的转移概率为 P_{XY}，X，Y=A，B，C，D，求窃听者不被发现的概率。

2．在现实生活中寻找量子密码的思想，构造模型。

第11章 密码协议

11.1 引言

密码算法是密码技术的基本元素，它的作用是依靠密码协议来实现的。所谓协议，就是一系列的操作步骤和约定。如果没有安全的协议，理论上再安全的算法也不可能实现安全通信的目的。最简单的例子如：在使用对称密钥密码体制进行通信时，在一般不安全信道上直接传送密钥，那么保密目的将彻底失败。相反，如果采用公钥密码体制的方法，将对称密钥加密后在公开信道上传送，那么，保密要求就可以实现了。但这样是否就一定安全了呢？答案是否定的，因为公钥的传递又带来了新的问题。

通信安全的主要内容应该包括：通信过程的可靠性，通信内容的保密性、可鉴别性（完整性），通信对象的认证，通信内容的不可否认性，密钥管理和分配技术等等。

通信过程的可靠性是指通信系统的稳定性，通信过程的可实现性，即系统保证通信任务完成的能力。这主要有系统稳定性、信道安全性等内容，如计算机网络中的操作系统、路由器、服务器的安全性，无线通信中的抗干扰能力等。

通信内容的保密性是密码技术的基本职能。

通信内容的完整性鉴别是鉴别信息完整性，识别篡改等内容的技术。数字签名、散列函数、单向函数等都是实现其目标的基本密码算法。

通信对象的认证是对通信对象的确认过程，也称为身份认证，是最重要的密码协议之一，也是现代通信中应用最广泛的技术。使用的主要密码算法包括数字签名、零知识证明等。

密钥管理和分配是信息保密性、完整性、身份认证的核心。常常由认证中心（CA）或仲裁中心（TA）来实现，这也是通信安全中最脆弱的地方。

密码协议与具体的密码应用紧密结合，本章结合几个具体的应用介绍安全密码协议的基本原理和应用。

11.2 公证协议

在现实生活中，有很多事务都离不开公证，如打官司、订合同、分遗产、甚至玩游戏等。在这些活动中，律师、法官、公证人等充当可信任的第三方，执行公证、裁决等事务。

类似的事务如何在计算机网络中实现呢？要通过公证协议。公证协议分三种：仲裁协议、裁决协议和自动执行协议。

11.2.1 仲裁协议

仲裁协议是一种业务、仲裁一体的协议。业务的执行必须在仲裁人的协助下实现。

下面以买卖协议来阐述仲裁协议。

设 A 要向 B 出售房产，C 是 A，B 信任的律师，则协议的完成过程如下：

1）A 将房屋产权证交给 C；

2）B 将现金支票交给 A；

3）A 到银行兑现支票；

4）A 将兑现信息知会 C；

5）C 根据兑现情况，将房产证交给或不交给 B。

可见，C 几乎参与了买卖的整个过程。这种协议的缺点是代价昂贵。

在计算机网络世界中，实现这一协议可能比买卖协议更困难，因为：

1）计算机网络世界中，通信的双方是无会面的，也是互不相信的；

2）仲裁者可能是一个程序、一台计算机，通信的双方要共同认可一个仲裁者可能是困难的，建立一个仲裁者开销也是巨大的；

3）仲裁者会增加通信的时延；

4）仲裁者会成为攻击的对象，成为新的安全节点。

这些问题都是实现仲裁协议必须解决的。

11.2.2 裁决协议

仲裁协议中，公证人的代价是昂贵的。所以人们通常采用一种事后仲裁协议，这就是裁决协议。业务的进行不需要仲裁者参与，仲裁只是通信双方发生争端时才需要，所以称为裁决。

裁决协议中的裁决人类似于现实世界中的法官，只有发生了官司时，才请求法官的裁决。

裁决协议有两个子协议：业务子协议和裁决子协议。

业务子协议：

1）协议双方 A，B 商定合同内容；

2）A，B 在合同上签名。

裁决子协议：

1）A，B 向裁决人 C 提出裁决要求；

2）A，B 提供证据；

3）C 根据证据裁决。

11.2.3 自动执行协议

自动执行协议是仲裁协议中最方便的协议，不需要仲裁者。在业务执行过程中，可以发现欺骗，并停止执行。

自动执行协议本质上是密码协议。数字签名就是一种自动执行协议。

仲裁协议，裁决协议，自动执行协议的对照见图 11-1、11-2、11-3。

11.3 密钥协议

密钥管理是信息安全中的关键，通常与具体应用和网络密切相关。结构上常常采用层次、分级加密管理方法。技术上通常采用公证中心，公钥密码算法、数字签名以及秘密共享等方

法；形式上通常有密钥分配中心（KDC）、公钥库等方案。

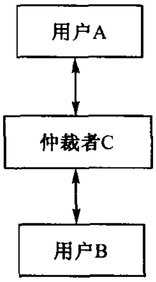


图 11-1 仲裁协议

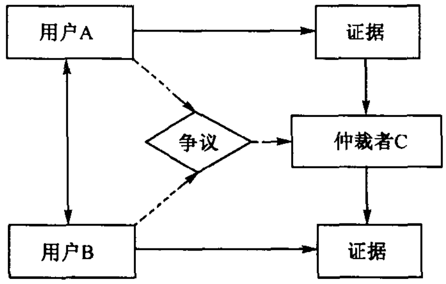


图 11-2 裁决协议

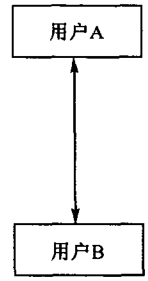


图 11-3 自动执行协议

下面介绍密钥管理中的最基本协议——密钥共享协议。

密钥共享协议是通信双方共享密钥的一系列方法，主要可分为密钥交换协议和密钥协商协议。

11.3.1 密钥交换协议

密钥交换是两个通信实体间传输密钥的过程。

本质上，密钥有对称密钥和公钥两种。

对称密钥的分发、传输需要密钥分配中心（KDC）的帮助。这一密钥传输过程有用户密钥注册和密钥传输两个阶段，如图 11-4，图 11-5 所示。

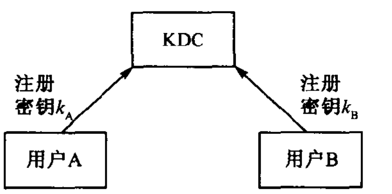


图 11-4 用户密钥注册

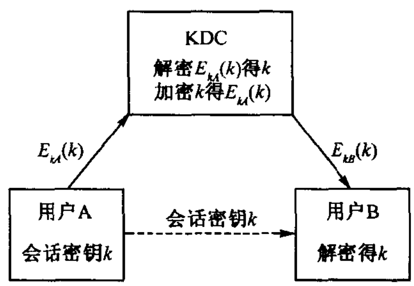


图 11-5 KDC 密钥传输

公开密钥传输借助于公钥库技术或公开索取。公钥库技术也分注册和传输两个阶段，如图 11-6 所示。

我们都知道，由于公钥密码体制算法速度慢，所以通常并不直接用公钥算法通信，而是用公钥算法传递对称密钥，最后用对称密码算法进行通信，这就是应用公钥密码体制实现的密钥传输。

但是这种方案的一个致命弱点是易遭受中间人攻击。设 A，B 需要建立会话密钥，C 窥知 A，B 的意图，当 B 向 A 索取公钥时，C 提供自己的公钥，声称自己是 A，于是，B 用 C 的公钥加密会话密钥，发送给 B，这时，C 截取并解密获取会话密钥；另一方面，C 向 A 索取公钥，并声称自己是 B，并将 B 的会话密钥加密后发送给 A。A 解密得到 B 的会话密钥，但 C 也得到了 A，B 的会话密钥。

中间人的攻击过程如图 11-7 所示。

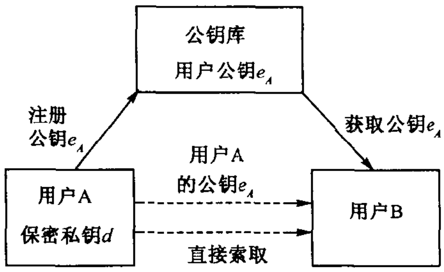


图 11-6 公钥的传输

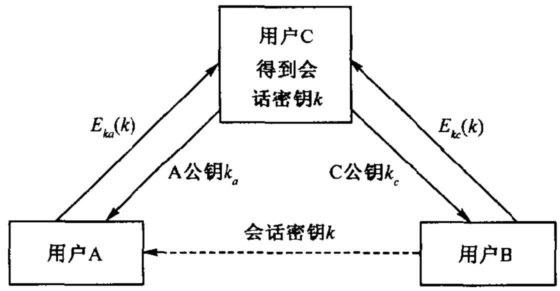


图 11-7 中间人攻击过程

中间人攻击的本质是：对 B 而言，C 充当 A；对 A 而言，C 充当 B。

中间人攻击能够实现的根本原因是没有身份认证，所以 A，B 都不能识别 C，使 C 有可乘之机。

对于公钥库方案，C 仍然可以适时偷换 A 的公钥，实现中间人攻击。但是，如果对公钥注册进行身份认证，就可以阻止中间人攻击。

数字签名也可以阻止中间人攻击，但是，数字签名的密钥传输又将带来新的攻击。

1984 年，R.Rivest 和 A.Shamir 提出了连锁协议，可以有效地阻止中间人攻击。

11.3.2 密钥协商协议

密钥协商是另一种实现密钥共享的方法。通信的双方没有主动、被动之分，而是双方通过一些数据交换各自生成密钥，但密钥是相同的。最典型的如 Diffie-Hellman 密钥交换协议、量子公钥分配协议及秘密共享方案等。

密钥管理涉及密钥产生、传输、使用和销毁等整个过程，是密码应用中的重要内容。除密钥共享协议外还有：密钥生成、密钥注册、密钥认证、密钥存储与备份、吊销以及密钥托管等内容。

11.4 秘密共享

秘密共享（Secrete Sharing）是密钥管理的一种核心技术，和其他密码技术一样，它源自现实世界，是现实世界在数字领域的映像。

以银行为例，银行的保险库是银行的核心重地，由谁来单独掌管都是不合适的，所以，一般都是交给几个人来共同掌管，例如，有 3 个出纳，规定需要 2 个出纳一起才能打开保险库。这一措施有两个优点：一是杜绝了一个人单独开库作案的隐患；二是 3 个人一共有 3 种开库的方法，防止了一个人遗忘密钥或缺席而打不开保险库的可能。这一设想实现了两个人共享一个秘密的思想。密码学把这种秘密共享方案称为门限方案。1979 年，A.Shamir 和 G.R.Blakley 首先提出这一思想，并各自给出了一严格方案。

首先介绍门限方案的严格定义：

定义 11-1 设 t，w（$t \leqslant w$）是正整数，如果在 w 个参与者中每人分发一个秘密数 k_i，

任何 t（或以上）人都能由 k_i 计算出密钥 k，而其中任何 $t-1$（或少于 $t-1$）人都不能计算出密钥 k，则称这一方案是 (w,t) 门限方案，其中的个人秘密数 k_i 称为子密钥或密钥 k 的影子。

11.4.1 Shamir 门限方案

Shamir 门限方案基于下面简单的数学原理：

一个 $t-1$ 次多项式，如果已知它在 t 个不同点的值，则可求出这个多项式（的所有系数）。

方案的详细描述如下：

设将被分割的（秘密）密钥是 k，p（$p>t$）是素数，密钥分发者构造 Z_p 中的一个 $t-1$ 次多项式：

$P_{t-1}(x)=a_0+a_1x+\cdots+a_{t-1}x^{t-1}$，并令 $a_0=k$，$a_i\in Z_p$，$i=1,\cdots,t-1$。

设有 w（$w>p$）个参与者，则分发者计算出该多项式在 w 个不同点 x_i 处的值 $y_i=P_{t-1}(x_i)$，并将点对 (x_i,y_i)（$i=1,\cdots,w$）分发给这 w 个参与者。多项式 $P_{t-1}(x)$ 保密，x_i 可以公开，也可以不公开。(x_i,y_i)（$i=1,\cdots,w$）就是分割出的子密钥，也称为密钥 k 的 w 个影子（密钥）。

如果有 t 个参与者会合，要计算密钥 k，不妨设他们的子密钥是：(x_i,y_i)（$i=1,\cdots,t$）。假设秘密多项式为：$P_{t-1}(x)=a_0+a_1x+\cdots+a_{t-1}x^{t-1}$，将 (x_i,y_i)（$i=1,\cdots,t$）代入得到 t 个方程：

$$\begin{cases}1\cdot a_0+x_1^1\cdot a_1+x_1^2\cdot a_2+\cdots+x_1^{t-1}\cdot a_{t-1}=y_1\\ 1\cdot a_0+x_2^1\cdot a_1+x_2^2\cdot a_2+\cdots+x_2^{t-1}\cdot a_{t-1}=y_2\\ \vdots\\ 1\cdot a_0+x_t^1\cdot a_1+x_t^2\cdot a_2+\cdots+x_t^{t-1}\cdot a_{t-1}=y_t\end{cases}$$

$a_0,a_1,\cdots,a_{t-1}$ 是所求系数，这是一个 t 个未知数，t 个方程的线性方程组，系数行列式为：

$\begin{vmatrix}1 & x_1^1 & x_1^2 & \cdots & x_1^{t-1}\\ 1 & x_2^1 & x_2^2 & \cdots & x_2^{t-1}\\ \vdots & \vdots & \vdots & & \vdots\\ 1 & x_t^1 & x_t^2 & \cdots & x_t^{t-1}\end{vmatrix}$，这是凡德蒙行列式，所以，当 $x_1,x_2,\cdots,x_t$ 互不相同时，其值非零，再根据线性方程组的克莱姆法则，知方程组有惟一解。不必求出所有系数，只要求出常数项即可。

$$k=a_0=\begin{vmatrix}1 & x_1^1 & x_1^2 & \cdots & x_1^{t-1}\\ 1 & x_2^1 & x_2^2 & \cdots & x_2^{t-1}\\ \vdots & \vdots & \vdots & & \vdots\\ 1 & x_t^1 & x_t^2 & \cdots & x_t^{t-1}\end{vmatrix}\Bigg/\begin{vmatrix}y & x_1^1 & x_1^2 & \cdots & x_1^{t-1}\\ y_2 & x_2^1 & x_2^2 & \cdots & x_2^{t-1}\\ \vdots & \vdots & \vdots & & \vdots\\ y_t & x_t^1 & x_t^2 & \cdots & x_t^{t-1}\end{vmatrix}。$$

可见 t 个或 t 个以上的参与者会合总能计算出密钥。但是少于 t 个参与者是不能计算出

密钥的。

由多项式的 t 个值求解多项式，也可以用 Lagrangre 插值公式来完成。如果已知多项式 $P_{t-1}(x)$ 的 t 个值 (x_i, y_i)（$i=1,\cdots,t$），则由 Lagrangre 插值公式：

$P_{t-1}(x)=\sum_{j=1}^{t}\left(\prod_{1\leqslant l\leqslant t,l\neq j}\frac{x-x_l}{x_j-x_l}\right)y_j$，所以，常数项为：

$k=P_{t-1}(0)=\sum_{j=1}^{t}\left(\prod_{1\leqslant l\leqslant t,l\neq j}\frac{x_l}{x_l-x_j}\right)y_j$，令：$b_j=\prod_{1\leqslant l\leqslant t,l\neq j}\frac{x_l}{x_l-x_j}$，则有：

$k=\sum_{j=1}^{t}b_j y_j$。可见，密钥 k 是 t 个子密钥 y_j 的线性函数。

特别地，如果 $w=t$，则 b_j 确定，所以可将 $z_j=b_j y_j$ 作为子密钥分发给参与者。那么，计算密钥的过程简化为：$k=\sum_{j=1}^{t}z_j$。

事实上，一个 (t,t) 门限方案，不须从多项式出发，直接由背包问题即可构造。

如设有 t 个参与者，密钥为 k，则随机地选取 t 个值 $z_1,\cdots,z_t$，满足：$z_1+\cdots+z_t=k$，将 $z_1,\cdots,z_t$ 分发给 t 个参与者即可。对 $z_1,\cdots,z_t$ 的选取，首先可任意选取 $z_1,\cdots,z_{t-1}$，最后再令 $z_t=k-(z_1+\cdots+z_{t-1})$。这时，计算密钥就显得更加简单了。

【例 11-1】 设有 7 个参与者，秘钥 $k=14$，秘钥分割者选取素数 $p=17$，Z_p 中多项式：

$P_4(x)=k+3x+6x^2+5x^3+12x^4$。

令 $x=1,2,3,4,5,6,7$，计算得到 $y=6,4,11,12,8,16,1$。

可得到 7 个子密钥：(1,6)，(2,4)，(3,11)，(4,12)，(5,8)，(6,16)，(7,1)，这就是一个（7,5）秘密共享门限方案。

假设持有子密钥（1,6)，(3,11)，(4,12)，(6,16)，(7,1) 的 5 个参与者要恢复密钥，则他们需要解线性方程组：

$$\begin{cases}k+a_1+a_2+a_3+a_4=6\\k+3a_1+3^2a_2+3^3a_3+3^4a_4=11\\k+4a_1+4^2a_2+4^3a_3+4^4a_4=12\\k+6a_1+6^2a_2+6^3a_3+6^4a_4=16\\k+7a_1+7^2a_2+7^3a_3+7^4a_4=1\end{cases}，即：\begin{cases}k+a_1+a_2+a_3+a_4=6\\k+3a_1+9a_2+10a_3+13a_4=11\\k+4a_1+16a_2+13a_3+1a_4=12\\k+6a_1+2a_2+12a_3+4a_4=16\\k+7a_1+15a_2+3a_3+4a_4=1\end{cases}。$$

解线性方程组，其系数行列式为：

$$D=\begin{vmatrix}1&1&1&1&1\\1&3&9&10&13\\1&4&16&13&1\\1&6&2&12&4\\1&7&15&3&4\end{vmatrix}=6，\quad D_0=\begin{vmatrix}6&1&1&1&1\\11&3&9&10&13\\12&4&16&13&1\\16&6&2&12&4\\1&7&15&3&4\end{vmatrix}=16。$$

在 Z_{17} 中，$D^{-1}=3$，所以，$k=3\times16\equiv14 \bmod 17$。这样就恢复了秘钥 14。

11.4.2 Blakley 秘密共享方案

Blakley 门限方案更加易懂，其数学原理如下：

设平面上有互不相同的 w 条直线交于同一点，则其中任意 2 条直线都可以决定这个交点，而任意单独的一条直线都不能决定这个交点。设交点的坐标（x，y）或 $x\|y$ 是密钥，每一条直线（的系数）是一个子密钥，w 个参与者的每一个人分配一条直线，则由此可作成一个（w,2）门限方案。

进一步，设 P_1，P_2，…，P_w 是 t（$t\leqslant w$）维空间中,交于同一点 k（密钥）的 w 个 t-1 维超平面，且其中任意 t 个超平面都可以决定这个点。将这 w 个超平面分发给参与者，每人一个，则得到一个（w,t）门限方案。

具体地，选取 w 个方程：

$$\begin{cases} a_{11}x_1+a_{12}x_2+\cdots+a_{1t}x_t=b_1 \\ a_{21}x_1+a_{22}x_2+\cdots+a_{2t}x_t=b_2 \\ \qquad\vdots \\ a_{w1}x_1+a_{w2}x_2+\cdots+a_{wt}x_t=b_w \end{cases},$$

记 $A=\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1t} \\ a_{21} & a_{22} & \cdots & a_{2t} \\ \vdots & \vdots & & \vdots \\ a_{w1} & a_{w2} & \cdots & a_{wt} \end{pmatrix}$，$B=\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_w \end{pmatrix}$，$X=\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_w \end{pmatrix}$，则方程组可写成：$AX=B$。记 $\tilde{A}=(A,B)$。

如果 $Rank(A)=Rank(\tilde{A})=t$，且对于 A 的任意 t 行所成矩阵 A_t，$Rank(A_t)=t$，

则方程组有惟一解，且其中任意 t 个方程也都有惟一解。

每一个参与者分发一个方程，即向量 $(a_{i1},a_{i2},\cdots,a_{it},b_i)$，$i$=1,2,…,$w$。

当 t 个参与者会合时，将得到 t 个未知数、t 个方程的方程组：

$$A_tX_t=B_t,$$

由于其系数矩阵的秩等于 t，所以得到惟一解，这也是整个方程组的解，也就是被分割的秘钥。

其实，可以看出，Shamir 的多项式方案也是一种线性方程组方案。无非是分发给参与者的子密钥（方程）由 $\left(1,x,x^2,x^3,\cdots,x^t,P_{t-1}(x)\right)$ 给出，比 Blakley 方案更具体。

11.4.3 带有不同权限的秘密共享方案

以上讨论的是一种简单的门限方案，它只能满足最普通的需要。如果参与者中有一些身份特殊者，他应该多掌握一些秘密。或者说，参与者中各人的访问权限不同，那么就必须修改门限方案，使其满足这些特殊需要。

更一般的秘密共享方案，称为访问结构，简单的数学描述如下：

设参与者的集合为 P，Γ 是 P 的某些子集的集合，满足：

对 $\forall B\in\Gamma$，如果 $B\subset A$，则 $A\in\Gamma$（单调性）；

则称 (P,Γ) 是一个访问结构。Γ 中元（P 的子集）称为授权集。

如果 $B\in\Gamma$，且对 $\forall C\subset B$，一定有 $C\notin\Gamma$，则称 B 为最小授权集。记 Γ 的最小授权集全体为 Γ_0，则称其为最小授权集类。

例如，在 Shamir (w,t) 门限方案中，$\Gamma=\{B\subset P:|B|\geqslant t\}$，$\Gamma_0=\{B\subset P:|B|=t\}$。

$|X|$ 表示 X 中元素的个数。

应用 (w,t) 门限方案可以很容易地实现具有不同访问权限的参与者的访问结构。

设参与者每一个人都有一个访问权限，用正整数来表示，设为：$n_i(i=1,\cdots,w)$。所谓具有不同访问权限的门限方案为：如果某些参与者的权限之和≥t，则他们可计算出密钥，否则不可以计算机出密钥。

设计一个具有不同访问权限的门限方案方法为：

计算 $n_1+\cdots+n_w=n$，设计一个 (n,t) 门限方案，将密钥分割成 n 个子密钥，对参与者，根据其权限 n_i，分配 n_i 个子密钥。则不同权限的访问即可实现。

11.4.4 秘密共享中的欺骗

以上的秘密共享方案仅仅解决了基本的秘钥分割问题，不能阻止欺骗，也不能满足进一步的要求。例如：

分割者的欺骗：秘钥分割者可以给某个参与者一个假的子密钥，使其不能享受应有的权利，且还被蒙在鼓里。

参与者的欺骗：某个参与者为了破坏一次会合解密，使用一个假子密钥参与恢复密钥，他不但破坏了这次行动，而且不会暴露。

冒充者的欺骗：某个骗子冒充自己拥有一个子密钥而参与恢复密钥，企图盗取他人的子密钥，但其他人却不能识别骗子，而只能互相猜测。

和其他密码技术一样，这些欺骗大都是方案本身的缺陷，是可以通过修正进行弥补的。于是，人们提出了满足各种要求的秘密共享方案。例如没有分割者的秘密共享方案，不暴露共享的秘密共享方案，可以验证子密钥有效性的秘密共享方案，可以添加、删除子密钥的秘密共享方案等等。

11.5 网络游戏

11.5.1 扑克游戏

计算机网络已经成为人们日常生活中不可缺少的一部分，有人甚至把计算机网络所构造的环境称为"赛博空间"、"网络空间"、"网络世界"等等。作为一个世界，一个空间，它确实已经具备了现实物质世界的许多要素，如：电子银行、电子商务、网上旅游、网络交友、网络游戏等等，甚至比物质世界更具活力。密码技术在电子银行、电子商务中的应用已成为密码应用的重要内容。密码技术在网络环境中的应用形式多样，网络游戏就是密码技术的一种典型应用，而网络扑克游戏是其中最普遍的一种。

日常生活中，几个人凑到一起玩玩扑克是一件既困难又容易的事。说困难是指一般很难凑齐那么多人，说容易是因为只要有牌，无须裁判都可以进行。

在网络环境中如何实现扑克游戏呢？参照现实世界，首先要解决发牌问题，其次是评判问题，对于网络世界来说，这是困难的。但是，这将寻找牌友的圈子扩大到了整个网络世界，各种水平的，操各种语言的，用各种约定的都有，无论是白天还是黑夜，都很容易开战，这就是网络世界的魅力。

1979 年，A.Shamir，R.Rivest 和 L.Adlman 首次提出了智力扑克（mental poker）的概念，他们认为，网络扑克要达到公正合理，需要满足如下条件：

（1）每一个参赛人员都知道自己手中的牌，但不知道别人的牌；

（2）发出的牌是符合牌理的（如：没有两张红心 A 等等）；

（3）参赛人员获取任何一张牌的概率都是一样的；

（4）比赛结束后可以验证比赛过程。

A.Shamir，R.Rivest 和 L.Adlman 用可换密码算法实现了智力扑克协议，现称为 SRA 方案。下面介绍应用 SRA 方案实现的四人或多人扑克协议：

设参赛者为 A、B、C、D 四人，52 张牌，按顺序：梅花 A，2，…，10，J，Q，K，方块 A，1，2，…，K，红心 A，1，2，…，K，黑桃 A，1，2，…，K 用 PK=1,2,…，52 表示。

1）A 用自己的加密密钥加密 1,2,⋯,52，按随机的顺序排列后发送给 B，称 A 是洗牌者；

2）B 随机地选择 13 张，用自己的加密密钥加密后发送给 A，将其余的牌发送给 C，称 B 是第 1 发牌者；

3）C 随机地选择 13 张，用自己的加密密钥加密后发送给 A，将其余的牌发送给 D，称 C 是第 2 发牌者；

4）D 随机地选择 13 张，用自己的加密密钥加密后和其余 13 张牌，一起发送给 A，称 D 是第 3 发牌者；

5）A 用自己的解密密钥解密 D 送来的最后 13 张牌，得到自己的牌，并将 B，C，D 的牌解密后分别发送给 B，C，D。

6）B，C，D 用自己的解密密钥解密得到自己的 13 张牌，这就完成了发牌。

可以看到：

（1）A、B、C、D 的加、解密算法必须是可换的，即：

$$E_{k_X} D_{k_Y} = E_{k_X} D_{k_Y}\text{，}\quad X,Y = A,B,C,D\text{。}$$

（2）由于参与者都是不见面的，所以存在“错牌”欺骗。A 可以说，他得到的牌有四个“A”，而 B 也有四个“A”，那么谁是欺骗者呢？所以，出现“错牌”现象时，全体参与者必须公布各自的密钥和发牌过程中获取、分发的密文和自己的牌；

（3）由验证的需要可见，参与者每一轮的密钥都应该不同（至少“错牌”一次，就要改变密钥一次），称为一轮一密；

（4）一轮一密的要求不仅为了阻止“错牌”攻击，而且还出于阻止“记牌“的可能：如果发牌者不变，密钥不变，则第 1 发牌人可以识别出上一轮自己所获的牌，经过几轮，他基本上可以识别出所有的牌，因此，当发牌人再一次发牌时，他就可以选择他想要的任

意牌了。

如果没有“错牌”，参与者的密钥也不变，为了阻止“记牌”的可能，发牌者每一轮可以选择一个不同的随机数 t，洗牌时加密 $1+t, 2+t, \cdots, 52+t$，由于 t 不同，所以每次加密后的密文都是不同的，但 t 必须同时公布给所有人。

以上协议满足智力扑克的基本要求，但是不能避免洗牌人 A 和第 1 发牌人 B 联合欺骗 C，D。A 可以告诉 B 哪些牌是好牌，哪些牌是坏牌，甚至可以把自己的解密密钥告诉 B，这样 B 选择自己想要的牌。但由于 A 的牌是由 C，D 选择后剩余的，所以 A 不能得到想要的牌。

RSA 是一种可换加密算法，是智力扑克的可选算法。但是，人们发现，RSA 算法却可以暴露某些牌的信息，即使一轮一密也不能避免：

如果 x 是一个模 n 平方剩余，即存在 a，$x=a^2 \bmod n$，RSA 加密密钥为 e，则加密后的密文 c 满足：$c=x^e \bmod n=(a^e)^2 \bmod n$，也是平方剩余。当然这暴露了牌的信息。除非所有牌的编码都是平方剩余或全是非平方剩余。

11.5.2 电子掷币协议

现实世界中，掷币博弈是通过博弈者面对面来实现公正、公平的。那么如何通过计算机网络、电话等方式来实现无会面掷币博弈呢？

掷币的简单模式如下：有 A、B 两个对象，A 产生一个 0 或者 1，B 猜测这个数，或 A 产生一个整数，B 猜测这个数是奇数或偶数来决定胜负等等。

掷币博弈应该满足的基本 “公正性”条件是：

（1）B 有 50%的机会猜中，或者说 A、B 各有 50%的机会取胜；

（2）如果 B 猜中，则 A 不能欺骗 B；另一方面 B 也不能否认自己猜测的结果。

M.Blum 方案是一个基于大数因子分解困难性的掷币协议：

1）A 选择两个大的不同素数 p,q，计算 $n=pq$，并发送给 B；

2）B 选择一个 b，计算 $a=b^2 \bmod n$，发送 A；

3）由于 A 知道 p,q，所以可以求解二次同余式：$x^2 \equiv a \bmod n$，得到四个根：$\pm b$，$\pm x$，$b \not\equiv \pm x \bmod n$，A 选择其一 z 发送给 B；

4）如果 B 得到的是 $\pm x$，则 B 可以求 $\gcd(b+x,n)$ 得到 p 或 q，从而分解 n，如果 B 得到的是 $\pm b$，则 B 不能分解 n。

假设 A、B 对通信数据 n,a,z 不可否认，则若以 B 能否分解 n 来决定胜负，那么以上协议满足公正掷币的基本条件：

（1）由于 A 选中 $\pm b$，$\pm x$ 的概率是 50%，所以 B 可以分解 n 的概率为 50%。如果 A 发送给 B 的不是 $\pm b$，$\pm x$ 之一，那么，B 验证 $z^2 \equiv b \bmod n$ 即可鉴别；

（2）如果 B 分解了 n，显然 A 不能欺骗 B。如果 B 不能分解 n，B 只能认输。

但是，以上协议存在“故意认输”欺骗，B 可以说自己选择的 $b=z$，从而不能分解 n。

【例 11-2】

（1）设 A 选取素数 $p=82499$，$q=120823$，计算 $n=pq$，得 $n=9967776677$ 发送给 B；

（2）B 选择一个正整数 $b=967268322$，计算 $a=(b^2 \bmod n)=2857285341$，发送给 A；

（3）A 解二次同余式：$x^2 \equiv 2857285341 \bmod 9967776677$，为此解：

$x^2 \equiv 2857285341 \bmod 82499$，$x^2 \equiv 2857285341 \bmod 120823$，

得到：$x = a^{\frac{p+1}{4}} \bmod p = 80207 \bmod 82499$，$x = -a^{\frac{p+1}{4}} \bmod p = 40616 \bmod 82499$，

$x = a^{\frac{q+1}{4}} \bmod q = 32453 \bmod 120823$，$x = -a^{\frac{q+1}{4}} \bmod q = 50046 \bmod 120823$。

得到 4 个方程：

$$\begin{cases} x \equiv 80207 \bmod 82499 \\ x \equiv 32453 \bmod 120823 \end{cases}, \begin{cases} x \equiv 80207 \bmod 82499 \\ x \equiv 50046 \bmod 120823 \end{cases},$$

$$\begin{cases} x \equiv 40616 \bmod 82499 \\ x \equiv 32453 \bmod 120823 \end{cases}, \begin{cases} x \equiv 40616 \bmod 82499 \\ x \equiv 50046 \bmod 120823 \end{cases}。$$

应用中国剩余定理，可得到 4 个解：

x_1=2330876700，x_2=967268322，$x_3 = n - x_2$=9000508355，$x_4 = n - x_1$=7636899977。

于是，A 在{2330876700，7636899977，967268322，9000508355}中选择一个发送给 B。

1）如果 A 选择的是 7636899977 或 2330876700，则 B 不能分解 n；

2）如果 A 选择的是 967268322 或 9000508355，则 B 计算

gcd(967268322+7636899977, 9967776677)=120823，或

gcd(9000508355+7636899977, 9967776677)=82499。

11.6 电子投票协议

投票选举是现实生活中又一种常见的活动，电子投票是一种重要的密码协议，已被广泛地应用于网络环境之中。

一般地，作为公平选举，为防止欺骗，选举应该满足以下基本要求：

（1）只有合法选举人才能投票；

（2）每一个投票人只能投票一次；

（3）每一个投票人都能检测自己的投票是否已被统计；

（4）每一个人都不能知道他人的投票情况；

（5）任何人都不能修改他人的投票。

和其他密码协议一样，也存在多种电子投票协议，它们分别满足各种要求。例如：带中心记票机构 CTF（Central Tabulating Facility）的投票协议，无 CTF 投票协议，有否决权的投票协议等等。

1981 年，David Chaum 给出了一种基于 RSA 盲签名的电子投票协议。

该协议需要两个机构：中心合法机构 CLA（Central Legitimigation Agency），中心记票机构 CTF 主持，且 CLA 和 CTF 分别称为身份登记机构和选委会。

（1）合法投票人向身份登记机构进行身份登记，经审查后获取选举证 v_A，$A \in I$，I 是全体投票人的集合，$\{(v_A, A) | A \in I\}$是投票人登记表；

（2）选委会构造一个 RSA 公钥体制：$RSA(p,q,e,d)$，$n = pq$ 为模数，e 为公开密钥，d 为秘密密钥；

（3）设投票人 A 的投票为 m，他选取一个随机数 r，计算 $c=\left(r^e m \bmod n\right)$，向选委会投票 $\left(c,v_A,id_A\right)$，v_A 是 A 的选举证，id_A 是 A 的身份信息；

（4）选委会根据 $\left(v_A,id_A\right)$ 确定选票的合法性，计算 $T'=\left(c^d \bmod n\right)$，送还给 A；

（5）A 计算 $T=\left(r^{-1}T' \bmod n\right)=\left(m^d \bmod n\right)$；

（6）A 以匿名形式投票 T；

（7）选委会计算 $m=T^e$，统计选票 m；

（8）选委会公布记名加密选票表 $\left\{\left(c,v_A,id_A\right) \mid A\in I\right\}$，无记名选票表 $\left\{\left(T,m\right)\right\}$；

（9）投票人可以通过 T 识别自己的选票，其他人无法从加密选票确定选票的内容，也无法从无记名选票确定投票人。

实现这一协议的核心算法就是：$T'=\left(c^d \bmod n\right)$，即选委会对消息 m 的盲签名。

11.7 习题

1．现实生活中，常常将一个信物分割成几份，交给几个人保管。当几个信物碎片合成一个完整的信物时，可以作为凭证取得信任。说明这一方法的思想和安全性原理。

2．用英文字母设计一个（10，4）门限方案。

3．设秘密 k=13，构造一个 Shamir（9，4）门限方案。

4．为银行的保险柜设计一把机械锁，使用钥匙来开锁，使这把锁具有（5，3）门限功能。

11.8 实验题

1．实现一个微型电子投票软件。

2．实现一个智力扑克游戏。

参 考 文 献

1 潘承洞，潘承彪. 初等数论. 北京：北京大学出版社，2002

2 （美）Bruce Schneier 著. 应用密码学——协议、算法与 C 源程序. 吴世忠等译. 北京：机械工业出版社，2001

3 王育民，刘建伟. 通信网的安全——理论与技术. 西安：西安电子科技大学出版社，1999

4 （美）William Stallings 著. 密码编码学与网络安全：原理与实践杨明等译. 第 2 版. 北京：电子工业出版社，2001

5 卿斯汉. 密码学与计算机网络安全. 北京：清华大学出版社，2001

6 卢开澄. 计算机密码学——计算机网络中的数据保密与安全. 第 2 版. 北京：清华大学出版社，1998

7 陈鲁升，沈世镒. 现代密码学. 北京：科学出版社，2002

8 胡予濮，张玉清等. 对称密码学. 北京：机械工业出版社，2002

9 （德）Bauer E L，著 密码编码和密码分析原理与方法. 吴世忠等译. 北京：机械工业出版社，2001

10 J. Daemen and V. Rijmen, AES proposal: Rijndael, http://www.esat.kuleuven.ac.be/~rijmen /rijndael/. 1998.

11 NESSIE, NESSIE security report, http://www.esat.kuleuen.ac.be/nessie/reports/, October,21,2002.

12 Cylink. Nomination of Safer++ as Candidate Algorithm for NESSIE, http://www.cosic. esat.kuleuven. ac.be/nessie/workshop/submissions/. 2000

13 Jakob Jonsson&RSA Laboratories Europe, RC6 Block Cipher, http://www.cosic.esat. kuleuven.ac.be/ nessie/workshop/submissions/. 2000

14 P.S.L.M.Barreto&V.Rijmen, The KHAZAD legacy-level block cipher, http://www.cosic.esat. kuleuven. ac.be/nessie/workshop/submissions/. 2000

15 T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, IEEE Transactions on Information Theory, vol. IT-31, no. 4, pp. 469-472, July 1985.

16 J. Ho_stein, J. Pipher, J.H. Silverman, NTRU: A new high speed public key cryptosystem, in Algorithmic Number Theory （ANTS III）, LNCS 1423, pp. 267–288, 1998.

17 RSA Laboratories. PKCS #1: RSA Cryptography Standard. Version 2.1, Jun,14,2002.

18 RSA Laboratories. PKCS #3: Diffie-Hellman Key Agreement Standard. Version 2.1, Jun,14,2002.

19 FIPS PUB 180-2,Secure Hash Standard, http://csrc.ncsl.nist.gov/fips/. NIST, 2001

20 A.Menezes. Ellptic curve public key cryposystems, Kluwer Academic Publishers,1993.

21 R.Schroppel,H.Ormaan,S.O'Malley and O.Spatacheck, Fast key exchange with elliptic curve systems, LNCS 963, D.Copersmith,Ed., Springer-Verlag, pp.43-56.1995

22 J.Solinas. An improved algorithm for arithmetic on a family of elliptic curves, LNCS 1294, B.Kaliski, pp. 357-371 Ed., Springer-Verlag, 1997.

23 IEEE P1363: Editorial Contribution to Standard for Public Key Cryptography, Februry 9,1998.

24 Julio Lopez and Ricardo Dahab, Improved algorithms for elliptic curve atithmetic in GF（2^n）, SAC'98 LNCS Springer-Verlag, 1998.

25 S.Wiesner. Conjugate coding Sigact News, Vol.15, no.1 1983, pp78-88. Original manuscript written

circa. 1970.

26 Townsed,P.D. Personal communication,July 1996.

27 Muller,A.H.Zbinden,N.Gisin Underwater quantum coding,Nature, pp 449, 1995, 378（30）

28 Brassard，Crépeau，Jozsa，Langlois. "A quantum bit commitment scheme provably unbreakable by both parties", In Proc. 34th IEEE Symp. on Foundations of Comp. Science. IEEE, pp.362-371. 1993.

29 A.Shamir . How to Share a Secrete? ,Communications of the ACM, v24. n11, pp.612-613. Nov, 1979.

30 G.R.Blakley . Safeguarding Cryptographic Keys, Proceedings of the National Computer Conference, 1979, American Federation of Imformation Proceesing Societies, pp.313-317. v48,1979.

［Ｇｅｎｅｒａｌ Ｉｎｆｏｒｍａｔｉｏｎ］
书名＝应用密码学
王衍波，薛通编著．２００３＿ｃ＿
作者＝
页数＝２１３
ＳＳ号＝
ＤＸ号＝
出版日期＝
出版社＝

书名
前言
目录
目录

出版说明
前言
第１章
初等数论基础
１．１
素数与因式分解
１．２
同余式理论
１．３

Ｅｕｌｅｒ定理
１．４
平方剩余
１．５
素性检验与模幂算法
１．６
指数与原根
１．７
习题
１．８
实验题
第２章

近世代数基础
2．1 群论初步
2．2 域论初步
2．3 有限域中的计算
2．4 习题
2．5 实验题
第3章 密码学基本概念

3．1 密码技术发展简介
3．2 密码系统的概念
3．3 密码分析
3．4 数字签名与认证
3．5 计算复杂性理论
3．6 传统密码举例

3.7 习题
第4章 对称密码算法
4.1 美国数据加密标准DES
4.1.1 历史
4.1.2 算法
4.1.3 DES

实例
4．1．4　DES安全性
4．1．5　DES工作方式
4．2　国际数据加密算法IDEA
4．2．1　IDEA加密算法

4.2.2 IDEA子密钥生成
4.2.3 IDEA解密
4.2.4 IDEA讨论
4.2.5 IDEA实例
4.3 美国高级数据加密标准AES

4
.3.1 算法
4
.3.2 基本运算
4
.3.3 基本部件
4
.3.4 AES
解密原理
4
.3.5 密钥生
成器
4

. 3 . 6 AES
实例
4 . 4
欧洲密码标准
4
. 4 . 1 引言
4
. 4 . 2 Kha
zad
4
. 4 . 3 RC6
4
. 4 . 4 Saf

e r + + 加密算法
4 . 5
习题
4 . 6
实验题
第 5 章
公钥密码算法
5 . 1
引言
5
. 1 . 1 公钥密
码思想
5

.1.2 Diffie-Hellman密钥交换协议
5.2 RSA密码体制
5.2.1 体制描述
5.2.2 算法分析
5.2.3 安全性分析
5.2.4 RSA举例

5.3 Rabin公钥密码体制
5.3.1 体制描述
5.3.2 解密算法及原理
5.3.3 安全性与因子分解
5.3.4 错误明

文攻击
5.3.5 示例
5.4 ElGamal公钥密码体制
5.4.1 体制描述
5.4.2 示例
5.5 MH背包公钥密码体制

5
. 5 . 1 0 - 1
背包问题
5
. 5 . 2 超递增序
列
5
. 5 . 3 背包密码
体制的原理
5
. 5 . 4 MH背包
密码体制的描述
5

.5.5 示例
5.6 概率公钥密码体制
5.6.1 引言
5.6.2 体制的描述
5.6.3 示例
5.7 NTRU公钥密码体制
5

. 7 . 1 NTRU的工作空间 5
. 7 . 2 密码体制 5
. 7 . 3 解密原理与参数选择 5
. 7 . 4 安全性分析 5
. 7 . 5 示例

5.8 公钥密码标准
5.8.1 PKCS#1
5.8.2 PKCS#3 v2.1
5.9 习题
5.10 实验题
第6章

数字签名方案与散列函数

6.1 数字签名方案
6.1.1 引言
6.1.2 RSA签名方案
6.1.3 ElGamal签名方案
6

．１．４　数字签名标准ＤＳＳ
６
．１．５　其他数字签名方案
６．２　散列函数
６
．２．１　签名与散列函数
６
．２．２　无碰撞散列函数

6.2.3 离散对数散列函数
6.2.4 散列函数的扩展
6.3 散列函数标准SHS
6.3.1 MD5
6.3.2 SHA

-1

6.3.3 SHA-256，SHA-384，SHA-512

6.3.4 MD5，SHA-1，SHA-256，SHA-384，SHA-512的比较

6.4

习题
6．5
实验题
第7章
序列密码
7．1
序列密码模型
7．2
序列的随机性概念
7．3
线性反馈移位寄存器
7．4

m序列及其随机性
7.5 周期序列的线性复杂度
7.6 习题
第8章 椭圆曲线密码学
8.1 引言
8.2 椭圆曲线的概念

8.2.1 概念和定义
8.2.2 群运算
8.2.3 椭圆曲线的分类
8.2.4 椭圆曲线分类加法公式
8.3 椭圆曲线群的结构

8
.3.1 群的阶
8
.3.2 椭圆曲线
的同构类
8.4
有限域F 2m上
的算术运算
8
.4.1 F 2m
的基及其元素的的运算
8
.4.2 有限域F

2n上椭圆曲线的倍点公式
8.5 椭圆曲线上的密码体制
8.5.1 Menezes-Vanstone密码体制
8.5.2 F2m上非超奇椭圆曲线Menezes-Va

nstone密码算法
8
.5.3 超奇椭圆曲线的密码体制
8
.5.4 EDSA——椭圆曲线数字签名标准ANSI X9.62
8
.5.5 其他体制的椭圆曲线版本
8.6

习题

第9章

混沌理论在密码学中的应用

9.1 混沌的基本概念

9.2 混沌序列的产生及其随机序列

9.3 逆混沌密码体制

9.4 示例

9.5 实验题
第10章 量子密码理论
10.1 引言
10.2 Heisenberg测不准原理
10.3 BB84协议
10.3.1 无窃

听量子信道的密钥分
配过程
1
０．３．２　有窃听
量子信道的密钥分配
过程
１０．
４　习题
第１１章
密码协议
１１．
１　引言
１１．

2　　公证协议
1
1．2．1　　仲裁协议
1
1．2．2　　裁决协议
1
1．2．3　　自动执行协议
11．3
密钥协议
1

1.3.1 密钥交换协议 1
1.3.2 密钥协商协议 1
1.4 秘密共享 1
1.4.1 Shamir门限方案 1
1.4.2 Blakley秘密共享方

案

1

1．4．3　带有不同权限的秘密共享方案

1

1．4．4　秘密共享中的欺骗

11．5　网络游戏

1

1．5．1　扑克游戏

11.5.2 电子掷币协议
11.6 电子投票协议
11.7 习题
11.8 实验题
参考文献